Author:
Fan, Rui

Title:
Real-Time Computer Stereo Vision for Automotive Applications

REAL-TIME COMPUTER STEREO VISION FOR AUTOMOTIVE APPLICATIONS

Rui Fan

Supervisors:
Dr. Naim Dahnoun
Prof. John G. Rarity

A dissertation submitted to the University of Bristol in accordance with the requirements for award of the degree of Doctor of Philosophy in the Faculty of Engineering

Department of Electrical & Electronic Engineering
University of Bristol

July 6, 2018

To all the wonderful time spent in UK between 2015 and 2018.

Abstract

Computer stereo vision technique has been prevalently used in various automotive applications for depth perception. This thesis mainly focuses on developing a computationally efficient and highly accurate disparity estimation algorithm for three automotive applications, i.e., lane detection, road surface 3-D reconstruction and pothole detection. Firstly, the real-time implementation of an efficient stereo matching algorithm is proposed to acquire the dense disparity maps for road scenes, where the road disparities decrease gradually from the bottom of the image to the top while the disparities of obstacles remain the same. Due to the fact that the disparities on the road surface change gradually, the disparities are estimated iteratively whereby the search range on each row is propagated from three estimated neighbouring disparities on the lower row. This not only minimises expensive computations but also reduces the ambiguities in stereo matching. The process of dense vanishing point estimation in a multiple lane detection system is then improved using the obtained disparity information. However, the outliers in the least squares fitting severely affect the accuracy when estimating the vanishing point. Therefore, the author uses Random Sample Consensus to update the parameters of the road model iteratively until the percentage of the inliers exceeds a pre-set threshold. This significantly helps the system to cope with some suddenly changing conditions. Furthermore, a novel lane position validation approach is proposed to compute the energy of each possible solution and select all satisfying lane positions for visualisation. The proposed lane detection algorithm is then implemented on a heterogeneous system for real-time purposes. Furthermore, the disparity estimation algorithm is used to reconstruct the 3-D model of a road surface for the purpose of condition assessment and damage detection. This is achieved by first transforming the perspective view of the target frame into the reference view, which not only increases the accuracy of the stereo matching for the road surface but also improves

the processing speed. Since the search range is obtained from the previous iteration, errors may occur when the propagated search is not sufficient. Therefore, a correlation maxima verification is performed to address this issue. To achieve the millimetre accuracy required for road condition assessment, a disparity map with subpixel resolution needs to be used. This is achieved by performing a parabola interpolation enhancement. Moreover, a novel disparity global refinement approach developed from Markov Random Fields and Fast Bilateral Stereo is introduced to further improve the accuracy of the estimated disparity map, where disparities are updated iteratively by minimising the energy function that is related to their interpolated correlation polynomials. Finally, the estimated dense subpixel disparity maps and the reconstructed 3-D point clouds are used in a pothole detection, classification and tracking system. The disparity map is first transformed to better distinguish the potholes from the road surface and a segmentation performed on the transformed disparity map can therefore separate the distress and non-distress areas accurately. To achieve a higher processing efficiency of the disparity map transformation, Golden Section Search and Dynamic Programming are utilised to estimate the transformation parameters efficiently. Then, a robust two-step disparity map modelling algorithm is proposed to fit a quadratic surface to the disparities in the non-distress area. The surface coefficients are coarsely estimated in the first step which are then updated iteratively in the following iterations to obtain fine estimates. The gradient information is also integrated into the process of surface fitting when determining the inliers and outliers in each iteration. Finally, different potholes are classified using Connected Component Labelling and the same pothole in successive frames is tracked using Discriminative Scale Space Tracking.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors Dr. Naim Dahnoun and Prof. John G. Rarity for their support and guidance throughout my Ph.D. programme.

I would also like to thank my colleagues and friends, especially Alexander Weiran Pang, Mohammud Junaid Bocus, Shuda Li, Xiao Ai, Chuanyu Yang, Umar Ozgunalp, Will Andrew, Brett Hosking and Yanan Liu for all their help and kind assistance and for making my stay in the United Kingdom really enjoyable over the last three years.

Furthermore, I would like to express my appreciation to Prof. Ioannis Pitas, Prof. David Bull, Prof. Roslyn Moran and Dr. Alin Achim for their useful suggestions and comments during the courses of my studies.

Last but not the least, my deepest gratitude goes to my parents, grandparents and wife Li Wang for their love, unconditional support and sincere prayers.

Author's Declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... **DATE: .....**

Contents

List of Figures

List of Acronyms

LIST OF FIGURES

Chapter 1

Introduction

1.1 Computer Stereo Vision

Humans live in a three-dimensional (3-D) world but our eyes can only perceive objects in two dimensions. The miracle of our depth perception is due to our brain's ability to analyse the difference between the two two-dimensional (2-D) images which are projected on the retinas of the eyes. In a broad sense, each pair of corresponding points from the retinas sends signals to the binocular neurons in the primary visual cortex [7]. The latter estimates the relative positional difference between each pair of corresponding points and this difference is generally known as *binocular disparity* [7].

As for the digital images captured by cameras, they are only 2-D in nature. In order to extrapolate the 3-D information from a given scene, multiple camera views of the same scene are required [8]. These images can be captured using either a single movable camera or an array of cameras [9]. In this thesis, the images from different viewpoints are captured using a binocular vision system which typically consists of a pair of synchronised digital cameras. The depth of the real-world scenery can thus be estimated by comparing the difference between the left and right digital images captured by each camera. This process is commonly referred to as *stereopsis* or *stereo vision*, and it is very similar to the human binocular vision [10].

1.1.1 The State-of-the-Art in Computer Stereo Vision

The two key aspects of computer stereo vision are speed and accuracy [11]. A lot of research has been carried out over the past decade to improve both the precision of disparity maps and the execution speed of the algorithm. However, the stereo vision algorithms which are designed to achieve better disparity accuracy usually have high computational complexity and low processing efficiency. Hence, speed and precision are two desirable but conflicting properties, and it is very challenging to achieve both of them simultaneously [12]. Therefore, the main motivation of investigating a stereo vision algorithm is to improve the trade-off between speed and accuracy. In most circumstances, a desirable trade-off entirely depends on the target application [13]. For instance, a real-time performance is required for the stereo vision systems applied in autonomous vehicles because the other entities of the systems, e.g., lane detection and obstacle detection, take up only a small portion of the processing time, and can be easily implemented in real-time if the 3-D information is available [13]. Although the algorithm execution can be improved with future advances in hardware computational power, the improvements performed on the algorithm side can further boost the processing speed [12].

The algorithms for disparity estimation can be classified as local, global and semi-global. Local algorithms simply match a series of blocks and select the correspondence with the lowest cost or the highest correlation. This optimisation is also known as winner-take-all (WTA). Unlike local algorithms, global algorithms process the stereo matching using some more sophisticated optimisation techniques, e.g., Graph Cut (GC) [14] and Belief Propagation (BP) [15]. These algorithms are commonly developed based on the Markov Random Fields (MRF) [16], where finding the best disparities is formulated as a probability maximisation problem. This is later addressed by energy minimisation approaches. Semi-Global Matching (SGM) [17] approximates the MRF inference by performing cost aggregation along all directions in the image and this greatly improves the accuracy and efficiency of stereo matching. However, the occlusion problem always makes it difficult to find the optimum value for the smoothness parameters: over-penalising the smoothness term can help avoid the ambiguities around dis-

continuities but on the other hand can lead to errors for continuous areas [18]. Therefore, some authors have proposed to break down the global problem into multiple local problems, each of which is affected by uncertainties to a lesser extent [19]. For instance, one alternative way of setting smoothness parameters is to group pixels in the image into different slanted planes [19–21]. Disparities in different plane groups are estimated with local constraints. However, this results in high computational complexities, making real-time performance challenging.

In order to further improve the trade-off between speed and accuracy, seed-and-grow local algorithms have been used extensively. In these algorithms, the disparity map is grown from a selection of seeds to minimise expensive computations and reduce mismatches caused by ambiguities. For example, the authors of [22–24] presented an efficient quasi-dense stereo matching algorithm, named Growing Correspondence Seeds (GCS), to estimate disparities iteratively with the search range propagated from a collection of reliable seeds. Similarly, various Delaunay Triangulation-Based Stereo Matching (DTSM) algorithms have been proposed in [25–27] to estimate tunable semi-dense disparity maps with the support of a piecewise planar mesh. The algorithm proposed in [11, 28] also provides an efficient strategy for local stereo matching whereby the search range is propagated from three estimated neighbouring disparities on the lower row. The algorithm [11] performs better than GCS and DTSM in terms of estimating dense disparity maps for road scenes where the road disparities decrease gradually from the bottom to the top, while the disparities of obstacles remain the same.

Furthermore, the deep neural networks have also been widely used in recent years to estimate the dense disparity maps [1, 29–31]. An example of training a deep neural network for stereo matching is illustrated in Figure 1.1. These approaches consider the stereo matching as a binary classification problem and learn a probability distribution over all disparity values [1]. When selecting a patch from the left image, the stereo matching algorithm predicts whether a corresponding patch in the right image is the correct match [31]. Although these approaches have achieved some impressive results on the benchmark of Middleburry [32–34] and KITTI [2, 35–38], the process of predicting the correct matches is computationally intensive and usually takes seconds or even minutes to execute on some state-of-the-art graphics cards, and therefore not suitable for real-time

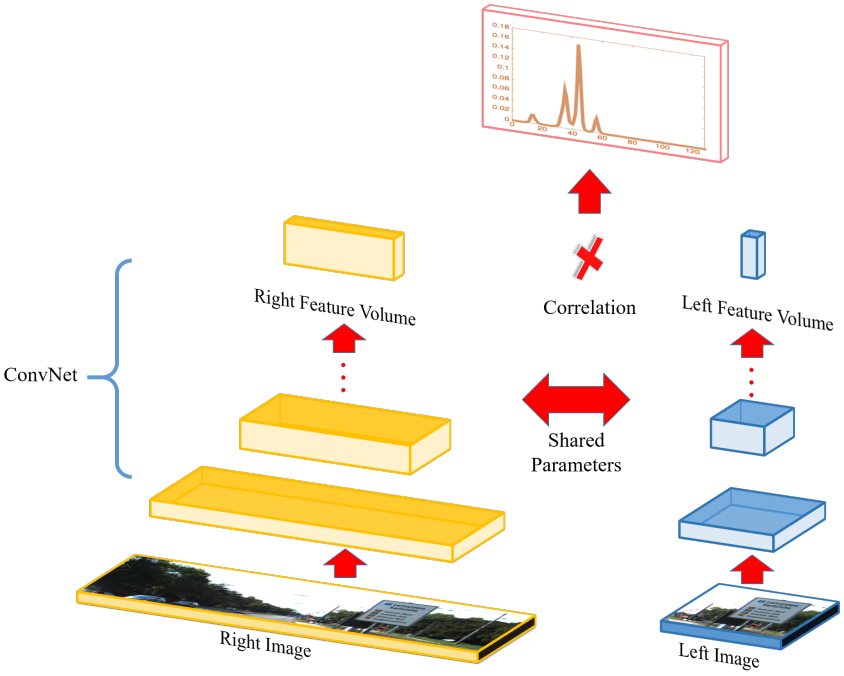


Figure 1.1: An example of deep learning for stereo matching [1].

applications.

1.1.2 Two Existing Problems in Computer Stereo Vision

To design an algorithm which is capable of providing highly accurate disparity maps, two problems in computer stereo vision must be tackled. The first problem is missing information, which is usually caused by occlusions or slanted surfaces [12]. These make some parts of a 3-D scene only visible in one image and therefore mismatches occur when estimating disparities. The other problem is the presence of homogeneous areas or repeated textures in an image [39]. The pixels inside a homogeneous area usually possess similar values which makes the stereo matching ambiguous. The same applies in an area with similar textures. To address the above problems, the stereo matching is usually processed using some more sophisticated optimisation techniques, e.g., GC and BP [40].

1.2 Automotive Applications

The deployment of autonomous vehicles has been increasing rapidly since Google first launched their self-driving car project in 2009 [41]. In recent years, with a number of technology breakthroughs being witnessed in the world where science fictions inventions are becoming a reality, the race to commercialise driver-less vehicles by many companies like Google, Tesla and BMW is fiercer than ever [11]. For instance, Volvo is planning to conduct self-driving experiments involving around 100 cars in China [42]. The driver-less vehicles are also able to communicate with each other via the 5G network which offers a higher speed internet access to transfer a large amount of data approximately 50 times faster than the current 4G systems. Furthermore, the computer stereo vision has also been prevalently used in prototype vehicle road tests to provide the depth information for various automotive applications, e.g., automotive vehicle navigation, simultaneous localisation and mapping and road condition assessment. An example of the fully equipped KITTI vehicle [2] is illustrated in Figure 1.2. The images from different viewpoints are captured simultaneously by the multi-camera system mounted on the KITTI vehicle. In this thesis, the automotive applications that the author focuses on include multiple lane detection, road surface 3-D reconstruction and pothole detection, classification and tracking. These will be discussed in the later

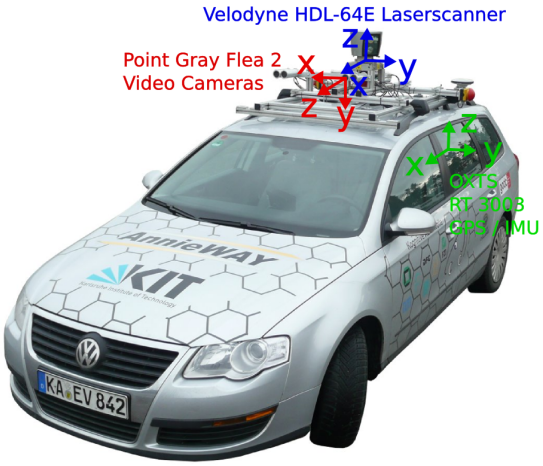


Figure 1.2: An example of the fully equipped KITTI vehicle [2].

chapters of this thesis.

1.3 Thesis Outline

- Chapter 2 provides some mathematical preliminaries for multiple view geometry and computer stereo vision. This chapter also provides a comprehensive literature review on lane detection, 3-D reconstruction and pothole detection. The heterogeneous system is briefly introduced at the end of this chapter.

- Chapter 3 describes an efficient stereo matching algorithm for automotive applications. The algorithm is implemented on a state-of-the-art graphics card for real-time purpose. The estimated disparity map is used as the input of the lane detection system presented in Chapter 4.

- Chapter 4 presents a real-time lane detection system based on dense vanishing point estimation. The disparity information acquired in Chapter 3 greatly helps to reduce redundant information and improve the robustness of vanishing point estimation for a non-flat road surface.

- Chapter 5 describes a road surface 3-D reconstruction algorithm based on dense subpixel disparity map estimation. The algorithm in this chapter is developed based on the stereo matching algorithm described in Chapter 3. The estimated disparity map and reconstructed 3-D point cloud are utilised as the inputs of the pothole detection, classification and tracking system described in Chapter 6.

- Chapter 6 presents a stereo vision-based pothole detection, classification and tracking system. The dense subpixel disparity map acquired in Chapter 3 serves as the input of this system. Different potholes are labelled with different colours while the same pothole in a sequence of successive frames is tracked with a scale adaptive tracker.

- Chapter 7 summarises the thesis and provides some recommendations for future work.

1.4 List of Publications and Submissions

1.4.1 Publications

- **R. Fan**, X. Ai, and N. Dahnoun, "Road Surface 3D Reconstruction Based on Dense Subpixel Disparity Map Estimation," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 3025-3035, 2018.

- **R. Fan**, N. Dahnoun, "Real-Time Stereo Vision-Based Lane Detection System," *Measurement Science and Technology*, 2018.

- U. Ozgunalp, **R. Fan**, X. Ai, and N. Dahnoun, "Multiple lane detection algorithm based on novel dense vanishing point estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 3, pp. 621-632, 2017.

- **R. Fan**, V. Prokhorov, and N. Dahnoun, "Faster-than-real-time linear lane detection implementation using SoC DSP TMS320C6678," in *Imaging Systems and Techniques (IST), 2016 IEEE International Conference on.* IEEE, 2016, pp. 306-311.

- **R. Fan**, and N. Dahnoun, "Real-time implementation of stereo vision based on optimised normalised cross-correlation and propagated search range on a GPU," in *Imaging Systems and Techniques (IST), 2017 IEEE International Conference on.* IEEE, 2017, pp. 1-6.

1.4.2 Submissions

- M. Evans, **R. Fan**, N. Dahnoun, "Iterative Roll Angle Estimation from Dense Disparity," *IEEE 7th Mediterranean Conference on Embedded Computing Resources.* (**accepted**)

- **R. Fan**, N. Dahnoun, "A Novel Disparity Map Transformation Algorithm Based on GSS and DP for Feature Extraction," *Information Processing Letters.* (Manuscript Number: IPL-D-18-00138)

- **R. Fan**, N. Dahnoun, "A Robust Pothole Detection, Classification and Tracking System Based on Stereo Vision Technique," *IEEE Transactions on Image Processing*. (Manuscript Number: TIP-18727-2018)

1.5 Open Source Projects and Demo Videos

The datasets and demo videos of some of the algorithms and systems described in the thesis have been made publicly available.

- The source code of the real-time computer stereo matching implementation presented in Chapter 3: https://drive.google.com/file/d/1hyi2-QsvdcUGl1e610LKDw5WRZ8WYcRo/view?usp=sharing
- The demo video of the real-time lane detection system described in Chapter 4: http://www.ruirangerfan.com/2018/02/multiple-lane-detection-algorithm-based_24.html
- The datasets and demo videos of the road surface 3-D reconstruction algorithm presented in Chapter 5: http://www.ruirangerfan.com/2017/07/road-surface-3d-reconstruction-based-on.html.
- The datasets of the robust pothole detection, classification and tracking system described in Chapter 6: http://www.ruirangerfan.com/2018/04/pothole-detection-classification-and.html.

Chapter 2

Background

In this chapter, the author details the background information for the readers. Firstly, some mathematical preliminaries are provided in Section 2.1. In Section 2.2 and Section 2.3, the author provides some basic but important concepts of multiple view geometry and computer stereo vision, respectively. The state-of-the-art lane detection algorithms are discussed in Section 2.4. The literature review of road surface 3-D reconstruction is given in Section 2.5. In Section 2.6, the author reviews the current pothole detection algorithms. Finally, some details on the heterogeneous system are provided in Section 2.7.

2.1 Preliminaries

2.1.1 Skew-Symmetric Matrix

In linear algebra, a *skew-symmetric matix* is defined as a square matrix $\boldsymbol{A}$ satisfying the condition that its transpose is equivalent to its negative, i.e., $\boldsymbol{A}^\top = -\boldsymbol{A}$ [43]. In 3-D computer vision, the skew-symmetric matrix $[\boldsymbol{a}]_\times$ of a vector $\boldsymbol{a} = [a_1, a_2, a_3]^\top$ is formulated as [8]:

$$[\boldsymbol{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{2.1}$$

The cross product of two vectors $\boldsymbol{a} = [a_1, a_2, a_3]^\top$ and $\boldsymbol{b} = [b_1, b_2, b_3]^\top$ is

equivalent to a matrix multiplication [8]:

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_{\times}\boldsymbol{b} = -[\boldsymbol{b}]_{\times}\boldsymbol{a} \tag{2.2}$$

Here, the author introduces two important properties of the skew-symmetric matrix, as shown in Eq. 2.3 and Eq. 2.4, where $\mathbf{0} = [0, 0, 0]^{\top}$ is a zero vector. These two properties are generally used for simplifying equations related to vector cross product.

$$\boldsymbol{a}^{\top}[\boldsymbol{a}]_{\times} = \mathbf{0}^{\top} \tag{2.3}$$

$$[\boldsymbol{a}]_{\times}\boldsymbol{a} = \mathbf{0} \tag{2.4}$$

2.1.2 Lie group SO(3) and SE(3)

In mathematics, a 3-D point $\boldsymbol{x_1} = [x_1, y_1, z_1]^{\top}$ can be transformed to another 3-D point $\boldsymbol{x_2} = [x_2, y_2, z_2]^{\top}$ using a rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ and a translation vector $\boldsymbol{t} \in \mathbb{R}^{3\times1}$ as follows:

$$\boldsymbol{x_2} = \boldsymbol{R}\boldsymbol{x_1} + \boldsymbol{t} \tag{2.5}$$

The rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ satisfies the conditions in Eq. 2.6, where $\boldsymbol{I}$ is an identity matrix and $\det(\boldsymbol{R})$ represents the determinant of $\boldsymbol{R}$. These rotation matrices are also known as *special orthogonal matrices*. The group containing all rotation matrices is denoted as a SO(3).

$$\boldsymbol{R}\boldsymbol{R}^{\top} = \boldsymbol{R}^{\top}\boldsymbol{R} = \boldsymbol{I}, \quad \det(\boldsymbol{R}) = 1 \tag{2.6}$$

The transformation from $\boldsymbol{x_1}$ to $\boldsymbol{x_2}$ can also be realised using the following equation:

$$\tilde{\boldsymbol{x}}_{\boldsymbol{2}} = \boldsymbol{P}\tilde{\boldsymbol{x}}_{\boldsymbol{1}} \tag{2.7}$$

where

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \tag{2.8}$$

$\tilde{\boldsymbol{x}}_{\boldsymbol{1}} = [\boldsymbol{x_1}^{\top}, 1]^{\top}$ and $\tilde{\boldsymbol{x}}_{\boldsymbol{2}} = [\boldsymbol{x_2}^{\top}, 1]^{\top}$ represent the homogeneous coordinates of $\boldsymbol{x_1}$

and $\boldsymbol{x_2}$, respectively. $\mathbf{0} = [0, 0, 0]^\top$ is a zero vector. $\boldsymbol{P}$ is a transformation matrix. The group containing all transformation matrices $\boldsymbol{P}$ is defined as a SE(3).

2.2 Multiple View Geometry

2.2.1 Perspective Camera Model

The perspective (or pinhole) camera model is the most commonly used geometric model to describe the relationship between a 3-D point $\boldsymbol{p}^{\mathscr{C}} = [X^{\mathscr{C}}, Y^{\mathscr{C}}, Z^{\mathscr{C}}]^\top$ in the Camera Coordinate System (CCS) and its projection $\bar{\boldsymbol{p}} = [x, y, z]^\top$ on the image plane π [10]. An example of the perspective camera model is shown in Figure 2.1, where $\boldsymbol{o}^{\mathscr{C}}$ is the focus of the camera and the distance between π and $\boldsymbol{o}^{\mathscr{C}}$ is the focal length f. $\hat{\boldsymbol{p}} = [\frac{X^{\mathscr{C}}}{Z^{\mathscr{C}}}, \frac{Y^{\mathscr{C}}}{Z^{\mathscr{C}}}, 1]^\top$ is an image point expressed in normalised coordinates. The ray originating from $\boldsymbol{o}^{\mathscr{C}}$ and going perpendicularly through π is known as the optical axis. $\boldsymbol{o} = [o_u, o_v]^\top$, and the intersection between π and the optical axis is the principal point in pixels. Since the distance z between $\boldsymbol{o}^{\mathscr{C}}$ and the image plane π is always equal to the focal length f, the relationship between $\boldsymbol{p}^{\mathscr{C}}$ and $\bar{\boldsymbol{p}}$ is shown as [10]:

$$\bar{\boldsymbol{p}} = \frac{f}{Z^{\mathscr{C}}} \boldsymbol{p}^{\mathscr{C}} \tag{2.9}$$

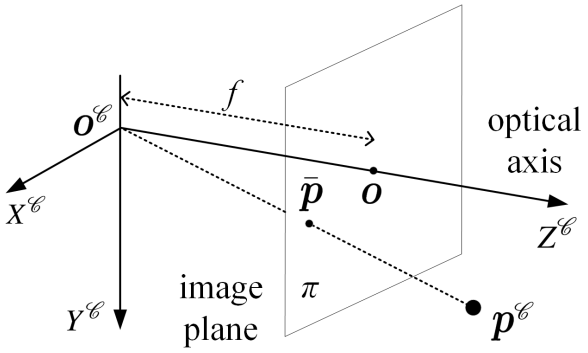


Figure 2.1: Perspective camera model.

2.2.2 Intrinsic Parameters

Since the lens distortion does not exist in a perspective camera model, the transformation from a projection point $\bar{\boldsymbol{p}}$ on the image plane to a pixel $\boldsymbol{p} = [u, v]^\top$ in the Image Coordinate System (ICS) is performed as follows:

$$\begin{aligned} u &= o_u + s_x x \\ v &= o_v + s_y y \end{aligned} \tag{2.10}$$

where s_x and s_y are the effective size measured in pixels per millimetre in the horizontal and vertical directions, respectively [10]. To simplify the expression of the intrinsic matrix $\boldsymbol{K}$, two notations $f_x = f s_x$ and $f_y = f s_y$ are introduced. Combining Eq. 2.9 and Eq. 2.10, a 3-D point $\boldsymbol{p}^{\mathscr{C}}$ in the CCS can be transformed to a pixel $\boldsymbol{p}$ in the ICS using Eq. 2.11, where $\tilde{\boldsymbol{p}} = [\boldsymbol{p}^\top, 1]^\top = [u, v, 1]^\top$ denotes the homogeneous coordinate of $\boldsymbol{p} = [u, v]^\top$. It is to be noted that an arbitrary 3-D point lying on the ray which goes from $\boldsymbol{o}^{\mathscr{C}}$ and through $\boldsymbol{p}^{\mathscr{C}}$ is always projected at $\boldsymbol{p}$ in the ICS. o_u, o_v, f, s_x and s_y [8] are five intrinsic parameters.

$$Z\tilde{\boldsymbol{p}} = \boldsymbol{K}\boldsymbol{p}^{\mathscr{C}} = \begin{bmatrix} f_x & 0 & o_u \\ 0 & f_y & o_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X^{\mathscr{C}} \\ Y^{\mathscr{C}} \\ Z^{\mathscr{C}} \end{bmatrix} \tag{2.11}$$

Given an intrinsic matrix $\boldsymbol{K}$, an image point can be expressed in normalised coordinates as follows [8]:

$$\hat{\boldsymbol{p}} = \boldsymbol{K}^{-1}\tilde{\boldsymbol{p}} = \frac{\bar{\boldsymbol{p}}}{f} = \frac{\boldsymbol{p}^{\mathscr{C}}}{Z^{\mathscr{C}}} \tag{2.12}$$

2.2.3 Lens Distortion

To get a better imaging result, a lens is usually installed in front of the camera but it introduces distortions into the images. The optical aberration caused by the lens deforms the physically straight lines and makes them appear as curves in the images. The lens distortions can be grouped into two main categories: radial and tangential [44]. The presence of radial distortion is due to the fact that lens's geo-

metric shape affects the straight line transmission, while the tangential distortion occurs because the lens installed in front of the camera is not perfectly parallel to the image plane. In practical experiments, the image geometry is affected to a much higher extent with radial distortion than with tangential distortion, and therefore the latter is always neglected when correcting a distorted image. For the calibration of a multi-camera system, the correction of lens distortion is usually carried out before estimating the intrinsic and extrinsic parameters [45].

2.2.3.1 Radial Distortion

The radial distortion can mainly be classified as barrel and pincushion [46]. An example of these two types of distortions is shown in Figure 2.2. It can be observed that the radial distortions are symmetric about the image centre and the lines are no longer straight in the distorted images. In barrel distortion, the image magnification decreases with the distance from the image centre (lines curve outwards). This type of radial distortion is commonly applied in fish-eye lenses to produce wide-view panoramic images. In contrast to barrel distortion, the pincushion distortion pinches the image (lines curve inwards) and it is widely applied in telephoto lenses to eliminate the globe effects.

Fortunately, these two types of radial distortions can be corrected using Eq. 2.13 [46], where $\boldsymbol{p} = [u, v]^\top$ is a pixel in the distorted image and $\boldsymbol{p_{corrected}} = [u_{corrected}, v_{corrected}]^\top$ is the displacement of $\boldsymbol{p}$ in the corrected image. $r = ||\boldsymbol{p}-\boldsymbol{o}||_2$ is the distance from $\boldsymbol{p}$ to the image centre. k_1, k_2 and k_3 are three intrinsic parameters used for correcting the radial distortion, and they can be estimated

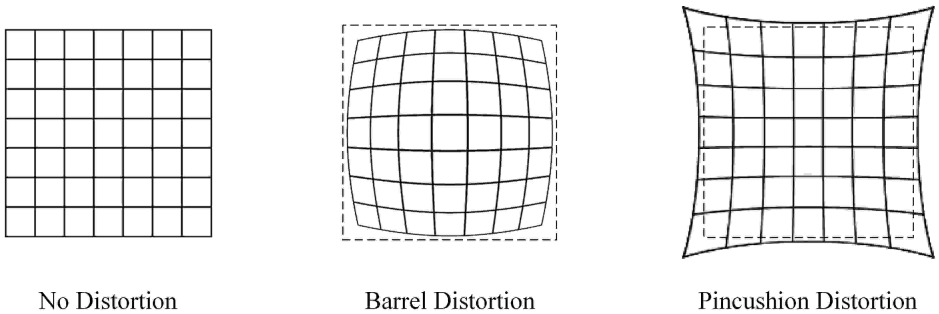


Figure 2.2: Radial distortion.

using several checkerboard images. An example of the distorted images containing a checkerboard pattern is shown in Figure 2.3a.

$$\begin{aligned} u_{corrected} &= (1 + k_1 r^2 + k_2 r^4 + k_3 r^6)(u - o_u) + o_u \\ v_{corrected} &= (1 + k_1 r^2 + k_2 r^4 + k_3 r^6)(v - o_v) + o_v \end{aligned} \tag{2.13}$$

2.2.3.2 Tangential Distortion

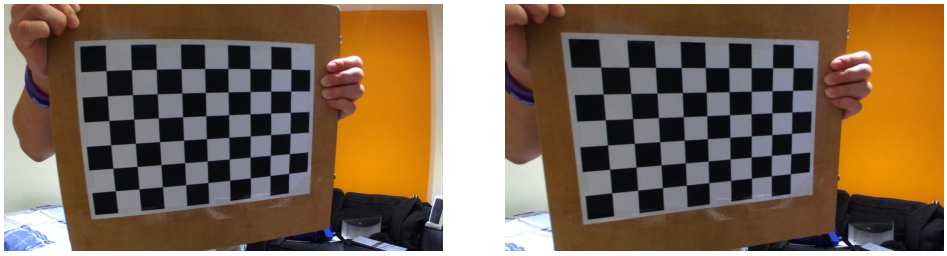
Similar to radial distortion, the tangential distortion can be corrected using Eq. 2.14 [47], where p_1 and p_2 are two intrinsic parameters which can be estimated using several images containing a planar checkerboard pattern.

$$\begin{aligned} u_{corrected} &= u + 2p_1(u - o_u)(v - o_v) + p_2(r^2 + 2(u - o_u)^2) \\ v_{corrected} &= v + p_1(r^2 + 2(v - o_v)^2) + 2p_2(u - o_u)(v - o_v) \end{aligned} \tag{2.14}$$

In practical experiments, radial and tangential distortions are usually corrected simultaneously [47]. The corresponding correction result of Figure 2.3a is shown in Figure 2.3b, where the bent checkerboard grids are now displayed linearly.

2.2.4 Epipolar Geometry

The generic geometry of a binocular vision system is known as *epipolar geometry* [10]. An example of epipolar geometry is shown in Figure 2.4, where $\boldsymbol{o}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{o}_{\boldsymbol{r}}^{\mathscr{C}}$ denote the focuses of the left and right cameras, respectively. $\boldsymbol{p}^{\mathscr{W}} = [X^{\mathscr{W}}, Y^{\mathscr{W}}, Z^{\mathscr{W}}]^\top$ is a 3-D point in the World Coordinate System (WCS) and its representations in the Left Camera Coordinate System (LCCS) and Right Camera



(a) (b)

Figure 2.3: Correcting lens distortion. (a) distorted image. (b) corrected image.

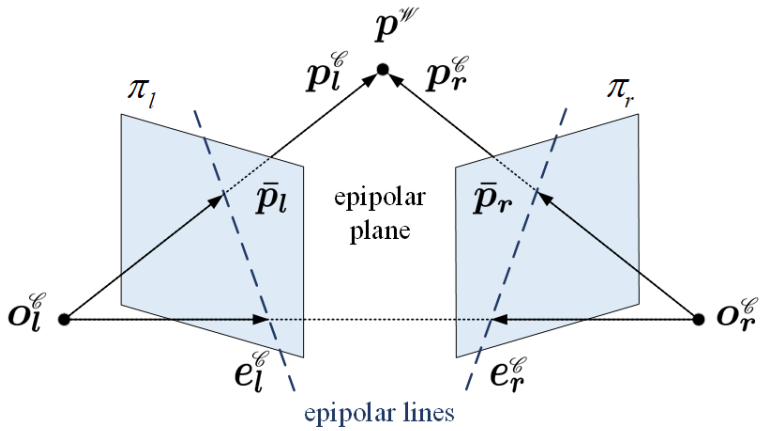


Figure 2.4: Epipolar geometry.

Coordinate System (RCCS) are $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} = [X_l^{\mathscr{C}}, Y_l^{\mathscr{C}}, Z_l^{\mathscr{C}}]^\top$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} = [X_r^{\mathscr{C}}, Y_r^{\mathscr{C}}, Z_r^{\mathscr{C}}]^\top$, respectively. π_l and π_r are two image planes. $\boldsymbol{p}^{\mathscr{W}}$ is projected on π_l at $\bar{\boldsymbol{p}}_{\boldsymbol{l}} = [x_l, y_l, f_l]^\top$ and on π_r at $\bar{\boldsymbol{p}}_{\boldsymbol{r}} = [x_r, y_r, f_r]^\top$, where f_l and f_r are the focuses of the left and right cameras, respectively. $\boldsymbol{e}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{e}_{\boldsymbol{r}}^{\mathscr{C}}$ denote the left and right epipoles, respectively. The plane identified by $\boldsymbol{o}_{\boldsymbol{l}}^{\mathscr{C}}$, $\boldsymbol{o}_{\boldsymbol{r}}^{\mathscr{C}}$ and $\boldsymbol{p}^{\mathscr{W}}$ is known as an *epipolar plane*. The latter intersects each plane in a line which is generally named as an *epipolar line*. Based on Eq. 2.9, the transformation from a 3-D point in each CCS to its projection on the corresponding camera plane can be performed as follows [10]:

$$\begin{aligned} \bar{\boldsymbol{p}}_{\boldsymbol{l}} &= \frac{f_l}{Z_l^{\mathscr{C}}} \boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} \\ \bar{\boldsymbol{p}}_{\boldsymbol{r}} &= \frac{f_r}{Z_r^{\mathscr{C}}} \boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} \end{aligned} \tag{2.15}$$

Similar to Eq. 2.12, $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$ can be normalised using Eq. 2.16, where $\boldsymbol{K}_{\boldsymbol{l}}$ and $\boldsymbol{K}_{\boldsymbol{r}}$ denote the intrinsic matrices of the left and right cameras, respectively. $\tilde{\boldsymbol{p}}_{\boldsymbol{l}} = [\boldsymbol{p}_{\boldsymbol{l}}{}^\top, 1]^\top$ and $\tilde{\boldsymbol{p}}_{\boldsymbol{r}} = [\boldsymbol{p}_{\boldsymbol{r}}{}^\top, 1]^\top$ represent the homogeneous coordinates of the

2-D points $\boldsymbol{p_l} = [u_l, v_l]^\top$ and $\boldsymbol{p_r} = [u_r, v_r]^\top$, respectively.

$$\begin{aligned} \hat{\boldsymbol{p}}_{\boldsymbol{l}} &= \frac{\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}}{Z_l} = {\boldsymbol{K_l}}^{-1}\tilde{\boldsymbol{p}}_{\boldsymbol{l}} \\ \hat{\boldsymbol{p}}_{\boldsymbol{r}} &= \frac{\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}{Z_r} = {\boldsymbol{K_r}}^{-1}\tilde{\boldsymbol{p}}_{\boldsymbol{r}} \end{aligned} \tag{2.16}$$

Before going into more details on the extrinsic parameters of an epipolar geometry, the author introduces a SO(3) matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ and a translation vector $\boldsymbol{t} \in \mathbb{R}^{3\times1}$ to describe the transformation from $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ to $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$ [8]:

$$\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} = \boldsymbol{R}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} + \boldsymbol{t} \tag{2.17}$$

2.2.5 Extrinsic Parameters

2.2.5.1 Essential Matrix

The *essential matrix* $\boldsymbol{E} \in \mathbb{R}^{3\times3}$ was first introduced by Longuet-Higgins in 1981 to establish a link between the epipolar constraints and the extrinsic parameters of a stereo system [48]. To introduce the defining equation of $\boldsymbol{E}$, a simple way is to multiply ${\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times$ on both sides of Eq. 2.17, as follows:

$${\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} = {\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times(\boldsymbol{R}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} + \boldsymbol{t}) \tag{2.18}$$

Using Eq. 2.2, Eq. 2.18 can be rewritten as follows:

$$-{\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}]_\times\boldsymbol{t} = {\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times\boldsymbol{R}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} + {\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times\boldsymbol{t} \tag{2.19}$$

Applying Eq. 2.3 and Eq. 2.4 to Eq. 2.19 yields Eq. 2.20 where the essential matrix $\boldsymbol{E}$ is defined as: $\boldsymbol{E} = [\boldsymbol{t}]_\times\boldsymbol{R}$.

$${\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top[\boldsymbol{t}]_\times\boldsymbol{R}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} = {\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}}^\top\boldsymbol{E}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} = 0 \tag{2.20}$$

Using Eq. 2.16, where $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$ are substituted by $\hat{\boldsymbol{p}}_{\boldsymbol{l}}$ and $\hat{\boldsymbol{p}}_{\boldsymbol{r}}$ respectively

and replacing these in Eq. 2.20, a new equation can be formulated as follows:

$$\hat{\boldsymbol{p}}_{\boldsymbol{r}}{}^{\top}\boldsymbol{E}\hat{\boldsymbol{p}}_{\boldsymbol{l}} = 0 \tag{2.21}$$

Therefore, $\boldsymbol{E}$ depicts the relationship between each pair of normalised image points $\hat{\boldsymbol{p}}_{\boldsymbol{l}}$ and $\hat{\boldsymbol{p}}_{\boldsymbol{r}}$ lying on the same epipolar plane. It is important to note here that $\boldsymbol{E}$ has five degrees of freedom (both $\boldsymbol{R}$ and $\boldsymbol{t}$ have three degrees of freedom, but the overall scale ambiguity causes the degrees of freedom to be reduced by one) [8]. Hence in theory, $\boldsymbol{E}$ can be estimated with at least five pairs of $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$. However, due to the non-linearity of $\boldsymbol{E}$, its estimation using five pairs of correspondences is always intractable. Therefore, $\boldsymbol{E}$ is commonly estimated with at least eight pairs of $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$ [10]. The algorithm for estimating $\boldsymbol{E}$ will be discussed in Section 2.2.5.2.

2.2.5.2 Fundamental Matrix

As mentioned in Section 2.2.5.1, the essential matrix creates a link between each pair of corresponding 3-D points in the LCCS and RCCS. When the intrinsic matrix of each camera is known, the relationship between each pair of corresponding 2-D points $\boldsymbol{p}_{\boldsymbol{l}} = [u_l, v_l]^{\top}$ and $\boldsymbol{p}_{\boldsymbol{r}} = [u_r, v_r]^{\top}$ can also be established. This process relates to a so-called *fundamental matrix*. It can be thought of as a generalisation of the essential matrix where the assumption of calibrated cameras is removed [8]. Applying Eq. 2.16 to Eq. 2.21 yields Eq. 2.22, where the fundamental matrix $\boldsymbol{F} \in \mathbb{R}^{3\times3}$ is defined as: $\boldsymbol{F} = \boldsymbol{K_r}^{-\top}\boldsymbol{E}\boldsymbol{K_l}^{-1}$.

$$\tilde{\boldsymbol{p}}_{\boldsymbol{r}}{}^{\top}\boldsymbol{K_r}^{-\top}\boldsymbol{E}\boldsymbol{K_l}^{-1}\tilde{\boldsymbol{p}}_{\boldsymbol{l}} = \tilde{\boldsymbol{p}}_{\boldsymbol{r}}{}^{\top}\boldsymbol{F}\tilde{\boldsymbol{p}}_{\boldsymbol{l}} = 0 \tag{2.22}$$

$\boldsymbol{F}$ has seven degrees of freedom: five are from $\boldsymbol{E}$ and the other two are from $\boldsymbol{K_l}$ and $\boldsymbol{K_r}$ [8]. The most commonly used algorithm to estimate $\boldsymbol{E}$ and $\boldsymbol{F}$ is a so-called "eight point algorithm" which was introduced by Hartley in 1997 [49]. This algorithm is based on the scale invariance of $\boldsymbol{E}$ and $\boldsymbol{F}$, i.e., $\lambda_E \boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}\top}\boldsymbol{E}\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} = 0$ and $\lambda_F \tilde{\boldsymbol{p}}_{\boldsymbol{r}}{}^{\top}\boldsymbol{F}\tilde{\boldsymbol{p}}_{\boldsymbol{l}} = 0$, where $\lambda_E, \lambda_F \neq 0$. By setting one element in $\boldsymbol{E}$ and $\boldsymbol{F}$ to 1, there are eight unknown elements that need to be estimated and this can be done using at least eight pairs of correspondences. If the intrinsic matrices $\boldsymbol{K_l}$

and $\boldsymbol{K_r}$ of the two cameras are known, the eight point algorithm only needs to be carried out once to estimate either $\boldsymbol{E}$ or $\boldsymbol{F}$ because the other one can be easily worked out using the relationship between them.

2.2.5.3 Homograph Matrix

For an arbitrary 3-D point $\boldsymbol{p^{\mathscr{W}}} = [X^{\mathscr{W}}, Y^{\mathscr{W}}, Z^{\mathscr{W}}]^\top$ lying on a planar surface $\boldsymbol{n}^\top \boldsymbol{p^{\mathscr{W}}} + \beta = 0$, its projections $\boldsymbol{p_l} = [u_l, v_l]^\top$ and $\boldsymbol{p_r} = [u_r, v_r]^\top$ on the left and right images can be linked with a so-called *homograph matrix* $\boldsymbol{H} \in \mathbb{R}^{3\times3}$, where $\boldsymbol{n} = [n_1, n_2, n_3]^\top$ is the normal vector of the planar surface. The expression of the planar surface can be rearranged as follows:

$$\frac{\boldsymbol{n}^\top \boldsymbol{p^{\mathscr{W}}}}{\beta} = 1 \tag{2.23}$$

Assuming that the LCCS and the WCS are identical and applying Eq. 2.15 and Eq. 2.23 to Eq. 2.17, the following expression is obtained:

$$\tilde{\boldsymbol{p}}_{\boldsymbol{r}} = \boldsymbol{H}\tilde{\boldsymbol{p}}_{\boldsymbol{l}} \tag{2.24}$$

where

$$\boldsymbol{H} = \frac{Z_l^{\mathscr{C}}}{Z_r^{\mathscr{C}}} \boldsymbol{K_r} \big(\boldsymbol{R} - \frac{\boldsymbol{t}\boldsymbol{n}^\top}{\beta}\big) \boldsymbol{K_l}^{-1} \tag{2.25}$$

The homograph matrix $\boldsymbol{H}$ is generally used to distinguish obstacles from a planar surface. For a well-calibrated stereo system, $\boldsymbol{R}$, $\boldsymbol{t}$, $\boldsymbol{K_l}$ and $\boldsymbol{K_r}$ are already known and $Z_l^{\mathscr{C}}$ is always equal to $Z_r^{\mathscr{C}}$. Therefore, $\boldsymbol{H}$ can be obtained by estimating $\boldsymbol{n}$ and β. Generally, $\boldsymbol{H}$ can be estimated with at least four pairs of correspondences $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ [8].

2.3 Stereopsis

2.3.1 Stereo Rectification

When using a pair of pinhole cameras to acquire images from multiple views, the main task of 3-D reconstruction is to determine each pair of corresponding points

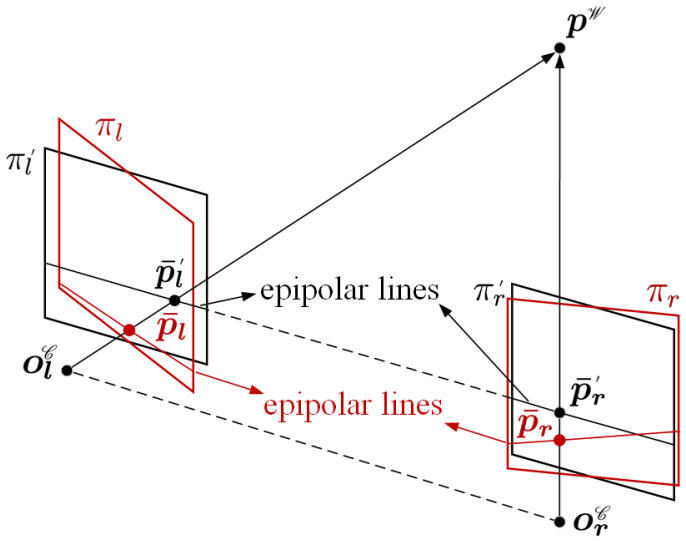


Figure 2.5: Stereo rectification.

between the left and right images. For an un-calibrated stereo vision system, finding the correspondence pairs usually involves a 2-D search, which is a computationally intensive task. Therefore, an image transformation process known as *stereo rectification* is always performed beforehand to reduce the dimension of the correspondence search. Each pair of conjugate epipolar lines becomes collinear and parallel to the horizontal image axis [10], as shown in Figure 2.5, where π_l and π_r are the original image planes and π_l' and π_r' are the rectified image planes. After the rectification process, the left and right images appear as if they were taken using a pair of parallel cameras. Hence, searching for the correspondence pairs is simplified to a one dimensional (1-D) process.

2.3.2 Basic Stereo Vision Model

A well-calibrated binocular system can be represented by a basic stereo vision model, as shown in Figure 2.6. The latter can be regarded as a specialisation of the epipolar geometry, where the left and right cameras are perfectly parallel to each other and the $X_l^{\mathscr{C}}$ axis and the $X_r^{\mathscr{C}}$ axis are collinear. $\boldsymbol{o}_l^{\mathscr{C}}$ and $\boldsymbol{o}_r^{\mathscr{C}}$ are the focus points of the left and right cameras, respectively. T_c, the baseline of the stereo rig, is defined as the distance between $\boldsymbol{o}_l^{\mathscr{C}}$ and $\boldsymbol{o}_r^{\mathscr{C}}$. The focal length of

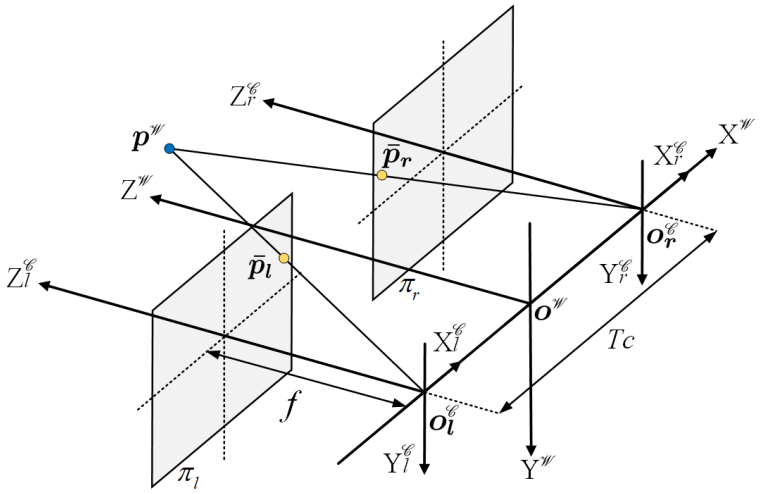


Figure 2.6: Basic stereo vision model.

each camera is denoted as f. $\boldsymbol{p}^{\mathscr{W}} = [X^{\mathscr{W}}, Y^{\mathscr{W}}, Z^{\mathscr{W}}]^\top$ is a point of interest in the WCS. Its representations in the LCCS and RCCS are $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} = [X_l^{\mathscr{C}}, Y_l^{\mathscr{C}}, Z_l^{\mathscr{C}}]^\top$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} = [X_r^{\mathscr{C}}, Y_r^{\mathscr{C}}, Z_r^{\mathscr{C}}]^\top$, respectively. $\boldsymbol{p}^{\mathscr{W}}$ is projected on π_l at $\bar{\boldsymbol{p}}_{\boldsymbol{l}} = [x_l, y_l, f]^\top$ and on π_r at $\bar{\boldsymbol{p}}_{\boldsymbol{r}} = [x_r, y_r, f]^\top$. $\boldsymbol{o}^{\mathscr{W}}$, the origin of the WCS, is at the centre of the line segment $L = \{t\boldsymbol{o}_{\boldsymbol{l}}^{\mathscr{C}} + (1-t)\boldsymbol{o}_{\boldsymbol{r}}^{\mathscr{C}} \mid t \in [0,1]\}$. The $Z^{\mathscr{W}}$ axis is perpendicular to π_l and π_r. Therefore, an arbitrary point $\boldsymbol{p}^{\mathscr{W}}$ in the WCS can be transformed to $\boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}}$ and $\boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}}$ as follows:

$$\begin{aligned} \boldsymbol{p}_{\boldsymbol{l}}^{\mathscr{C}} &= \boldsymbol{I}\boldsymbol{p}^{\mathscr{W}} + \boldsymbol{t}_{\boldsymbol{l}} \\ \boldsymbol{p}_{\boldsymbol{r}}^{\mathscr{C}} &= \boldsymbol{I}\boldsymbol{p}^{\mathscr{W}} + \boldsymbol{t}_{\boldsymbol{r}} \end{aligned} \tag{2.26}$$

where $\boldsymbol{t}_{\boldsymbol{l}} = [\frac{T_c}{2}, 0, 0]^\top$ and $\boldsymbol{t}_{\boldsymbol{r}} = [-\frac{T_c}{2}, 0, 0]^\top$. $\boldsymbol{I}$ is an identity matrix. Applying Eq. 2.15 to Eq. 2.26 results in the following expressions:

$$\begin{aligned} x_l &= f\frac{X^{\mathscr{W}} + T_c/2}{Z^{\mathscr{W}}}, \quad y_l = f\frac{Y^{\mathscr{W}}}{Z^{\mathscr{W}}} \\ x_r &= f\frac{X^{\mathscr{W}} - T_c/2}{Z^{\mathscr{W}}}, \quad y_r = f\frac{Y^{\mathscr{W}}}{Z^{\mathscr{W}}} \end{aligned} \tag{2.27}$$

Assuming $\boldsymbol{K}_{\boldsymbol{l}}$ and $\boldsymbol{K}_{\boldsymbol{r}}$ are identical and applying Eq. 2.27 to Eq. 2.10, the

following expressions are obtained:

$$\begin{aligned} \boldsymbol{p_l} &= \begin{bmatrix} u_l \\ v_l \end{bmatrix} = \begin{bmatrix} f_x \frac{X^{\mathscr{W}}}{Z^{\mathscr{W}}} + o_u + f_x \frac{T_c}{2Z^{\mathscr{W}}} \\ f_y \frac{Y^{\mathscr{W}}}{Z^{\mathscr{W}}} + o_v \end{bmatrix} \\ \boldsymbol{p_r} &= \begin{bmatrix} u_r \\ v_r \end{bmatrix} = \begin{bmatrix} f_x \frac{X^{\mathscr{W}}}{Z^{\mathscr{W}}} + o_u - f_x \frac{T_c}{2Z^{\mathscr{W}}} \\ f_y \frac{Y^{\mathscr{W}}}{Z^{\mathscr{W}}} + o_v \end{bmatrix} \end{aligned} \tag{2.28}$$

In general, s_x is usually assumed to be 1 and therefore $f_x = f_y = f$. The relationship between disparity d and depth $Z^{\mathscr{W}}$ is as follows [50]:

$$d = u_l - u_r = f\frac{T_c}{Z^{\mathscr{W}}} \tag{2.29}$$

According to Eq. 2.29, d is inversely proportional to $Z^{\mathscr{W}}$. Therefore, for a distant 3-D point $\boldsymbol{p}^{\mathscr{W}}$, the vertical coordinates of $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ are similar to each other. On the other hand, when $\boldsymbol{p}^{\mathscr{W}}$ is lying near the stereo rig, the difference between the vertical coordinates of $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ is large.

The depth error in the WCS is denoted as $\Delta Z^{\mathscr{W}}$. The disparity error Δd can be computed as follows:

$$\Delta d = f\frac{T_c}{Z^{\mathscr{W}}} - f\frac{T_c}{Z^{\mathscr{W}} + \Delta Z^{\mathscr{W}}} = \frac{fT_c\Delta Z^{\mathscr{W}}}{(Z^{\mathscr{W}} + \Delta Z^{\mathscr{W}})Z^{\mathscr{W}}} \tag{2.30}$$

Assuming that Δd is a constant, Eq. 2.30 can be rearranged as follows:

$$\Delta Z^{\mathscr{W}} = \frac{{Z^{\mathscr{W}}}^2\Delta d}{fT_c - Z^{\mathscr{W}}\Delta d} \tag{2.31}$$

Eq. 2.31 shows that the depth error $\Delta Z^{\mathscr{W}}$ is proportional to the depth $Z^{\mathscr{W}}$ but inversely proportional to the baseline T_c. Therefore, the reconstruction precision of a 3-D object decreases when it moves away from the stereo rig. This can however be improved by increasing the baseline of the stereo rig.

2.3.3 Disparity Estimation

As mentioned in Section 2.3.2, for a well-calibrated stereo vision system, finding the correspondence pairs only involves a 1-D search and therefore the disparity

map can be obtained by simply computing the horizontal distance between each pair of correspondences $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$. This process is generally referred to as *disparity estimation* or *stereo matching* [32, 40]. The latter usually produces a left disparity map ℓ^{lf} and a right disparity map ℓ^{rt}, as shown in Figure 2.7b and Figure 2.7d, respectively. The left disparity map ℓ^{lf} is obtained by assigning the left image (see Figure 2.7a) and the right image (see Figure 2.7c) as the reference and target, respectively. On the contrary, the right disparity map ℓ^{rt} can be obtained by setting the right image (see Figure 2.7c) as the reference, which is then compared with the left image (see Figure 2.7a). In general, a disparity estimation algorithm usually consists of the following four steps [32]:

1. cost computation
2. cost aggregation
3. disparity optimisation

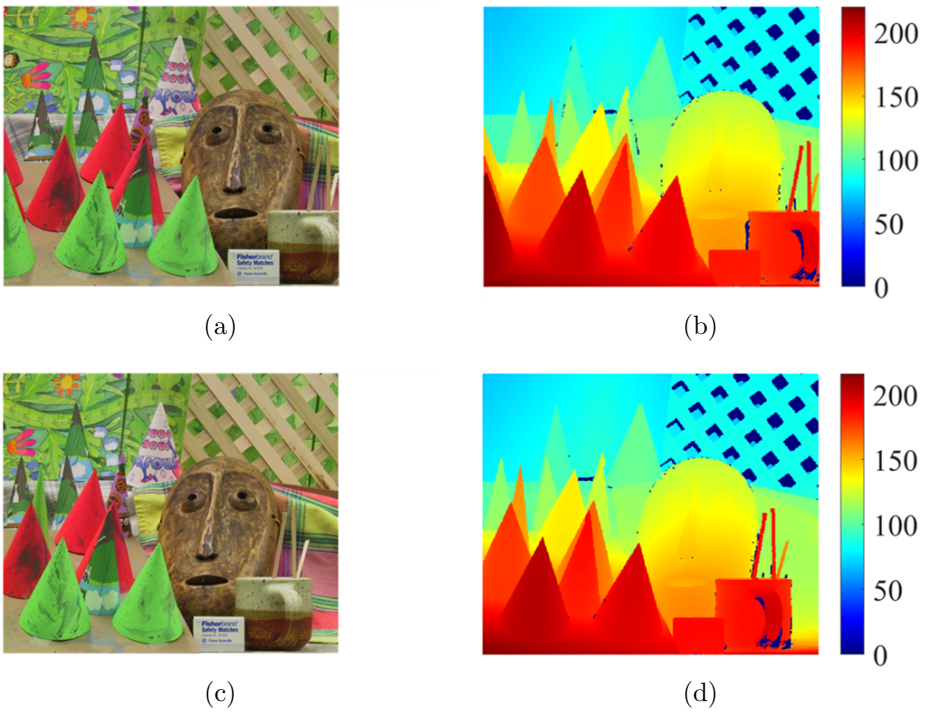


Figure 2.7: Left and right images and disparity maps. (a) left image. (b) left disparity map. (c) right image. (d) right disparity map.

4. disparity refinement

However, the sequential use of these four steps depends entirely on the chosen algorithm [32]. For instance, for most local block matching-based algorithms, the first and second steps are used together and the desirable disparity can be determined by simply finding the one which corresponds to the highest correlation or the lowest cost [11]. However, for most global algorithms, finding the desirable disparity is regarded as an optimisation problem which can be solved by minimising a global energy [40]. The last step usually involves some disparity map refinement techniques, e.g., Weighted Median Filtering, Left-Right Consistency (LRC) Check and Subpixel Enhancement.

2.3.3.1 Cost Computation

The disparity d is a random variable with N possible discrete states. Each possible state is associated with a cost c which can be computed using a cost function. The most common pixel-wise matching costs include the Absolute Difference (AD) c_{AD} and the Squared Difference (SD) c_{SD} [32]. These two types of matching costs can be computed using the cost functions as follows [51]:

$$c_{AD}(u, v, d) = |i_l(u, v) - i_r(u - d, v)| \tag{2.32}$$

$$c_{SD}(u, v, d) = (i_l(u, v) - i_r(u - d, v))^2 \tag{2.33}$$

where $i_l(u, v)$ denotes the intensity of a pixel located at (u, v) in the left image and $i_r(u - d, v)$ represents the pixel intensity at the position of $(u - d, v)$ in the right image. The left and right images are usually in gray-scale format. In global algorithms, the pixel-wise matching costs are used to analyse the compatibility between disparities and the corresponding intensity differences, while the local algorithms usually perform an aggregation of pixel-wise matching costs over all pixels within a certain neighbourhood [51].

2.3.3.2 Cost Aggregation

For local algorithms, the pixel-wise matching costs are usually aggregated over all pixels within a certain support region to minimise the incorrect matches [39]. Since the support regions are usually rectangular blocks, these local algorithms are also known as *stereo block matching* [11]. The general expression of the cost aggregation is as follows [32]:

$$c_{agg}(u, v, d) = w(u, v, d) * C(u, v, d) \tag{2.34}$$

where the centre of the support region is (u, v) and the corresponding disparity is d. w is a kernel that represents the support region. c_{agg} denotes the cost aggregation result and C represents a neighbourhood system containing the pixel-wise matching costs over all pixels within w. c_{agg} can be obtained by performing a convolution between w and C.

In general, square blocks are used as the support regions, and the convolution process can be regarded as a box filtering [32]. The aggregations of c_{AD} and c_{SD} within a square block are also known as the Sum of Absolute Difference (SAD) and the Sum of Squared Difference (SSD), respectively. The cost functions of the SAD and the SSD are as follows [39]:

$$c_{SAD}(u, v, d) = \sum_{x=u-\rho}^{x=u+\rho} \sum_{y=v-\rho}^{y=v+\rho} |i_l(x, y) - i_r(x - d, y))| \tag{2.35}$$

$$c_{SSD}(u, v, d) = \sum_{x=u-\rho}^{x=u+\rho} \sum_{y=v-\rho}^{y=v+\rho} (i_l(x, y) - i_r(x - d, y))^2 \tag{2.36}$$

where c_{SAD} and c_{SSD} represent the costs of the SAD and the SSD, respectively. The side length of the square block is $2\rho + 1$. Due to the fact that ρ is a positive integer number, the side length of the square block is always an odd number. The centres of the left and right blocks are (u, v) and (u, v, d), respectively.

Although the SAD and the SSD have low computational complexities, they are very sensitive to the intensity difference [39]. In this regard, some other cost functions, such as the Normalised Cross-Correlation (NCC), which are insensitive to the intensity difference, are more popular for local algorithms [39]. The cost

function of the NCC is as follows [11]:

$$c_{NCC}(u,v,d) = \frac{1}{n\sigma_l\sigma_r}\sum_{x=u-\rho}^{x=u+\rho}\sum_{y=v-\rho}^{y=v+\rho}(i_l(x,y)-\mu_l)(i_r(x-d,y)-\mu_r) \tag{2.37}$$

where

$$\sigma_l = \sqrt{\sum_{x=u-\rho}^{x=u+\rho}\sum_{y=v-\rho}^{y=v+\rho}(i_l(x,y)-\mu_l)^2/n} \tag{2.38}$$

$$\sigma_r = \sqrt{\sum_{x=u-\rho}^{x=u+\rho}\sum_{y=v-\rho}^{y=v+\rho}(i_r(x-d,y)-\mu_r)^2/n} \tag{2.39}$$

The cost $c_{NCC} \in [-1,1]$ reflects the similarity between each pair of left and right blocks and a higher value of c_{NCC} corresponds to a better matching. μ_l and μ_r represent the means of the pixel intensities within the left and right blocks, respectively. σ_l and σ_r denote the standard deviations of the left and right blocks, respectively. $n = (2\rho+1)^2$ represents the number of pixels within each block.

However, finding the correct disparity value is still a very intractable task when the block size is fixed [39]. For example, choosing a large block size can help reduce the ambiguities during the stereo matching but on the other hand can lead to errors in discontinuous areas [40]. Therefore, some authors proposed to aggregate the costs adaptively to improve the accuracy of the estimated disparity map [52].

Since Tomasi et al. introduced the bilateral filter in [53], many authors have investigated its applications to aggregate the matching costs adaptively [54–56]. These methods are also known as Fast Bilateral Stereo (FBS), where both intensity difference and spatial distance provide a weight to adaptively constrain the aggregation of discontinuities. A general representation of the cost aggregation in FBS is represented as follows:

$$c_{agg}(u,v,d) = \frac{\sum_{x=u-\rho}^{u+\rho}\sum_{y=v-\rho}^{v+\rho}\omega_d(x,y)\omega_r(x,y)c(x,y,d)}{\sum_{x=u-\rho}^{u+\rho}\sum_{y=v-\rho}^{v+\rho}\omega_d(x,y)\omega_r(x,y)} \tag{2.40}$$

where ω_d and ω_r are based on the spatial distance and the colour similarity. The costs c within a square block are aggregated adaptively to obtain c_{agg},

respectively.

Although the FBS has shown a good performance in terms of matching accuracy, it usually takes a long time to process the whole cost volume [40]. Therefore, it is usually implemented on powerful hardware to improve the trade-off between accuracy and speed.

2.3.3.3 Disparity Optimisation

For local algorithms, the disparity costs are calculated by shifting a series of blocks from the right image between d_{min} and d_{max} and matching them with a constant square block from the left image. The disparity with the lowest cost or the highest correlation is then selected as the correspondence. This optimisation is also known as WTA, where d_{min} and d_{max} are decided by the furthest and the closest objects, respectively.

Unlike the principle of WTA applied in local stereo matching algorithms, the matching costs from neighbouring pixels are also taken into account in global algorithms, e.g., GC and BP. The MRF is a commonly used graphical model in these global algorithms. An example of the MRF model is depicted in Figure 2.8.

The graph $\mathscr{G} = (\mathscr{P}, \mathscr{E})$ is a set of vertices $\mathscr{P}$ connected by edges $\mathscr{E}$, where $\mathscr{P} = \{\boldsymbol{p_{11}}, \boldsymbol{p_{12}}, \cdots, \boldsymbol{p_{mn}}\}$ and $\mathscr{E} = \{(\boldsymbol{p_{ij}}, \boldsymbol{p_{st}}) \mid \boldsymbol{p_{ij}}, \boldsymbol{p_{st}} \in \mathscr{P}\}$. Two edges sharing one common vertex are called a pair of adjacent edges [57]. Since the MRF is considered to be undirected, $(\boldsymbol{p_{ij}}, \boldsymbol{p_{st}})$ and $(\boldsymbol{p_{st}}, \boldsymbol{p_{ij}})$ refer to the same edge here. $\mathscr{N}_{ij} = \{\boldsymbol{n_{1p_{ij}}}, \boldsymbol{n_{2p_{ij}}}, \cdots, \boldsymbol{n_{kp_{ij}}} \mid \boldsymbol{n_{p_{ij}}} \in \mathscr{P}\}$ is a neighbourhood system for $\boldsymbol{p_{ij}}$.

For stereo vision problems, $\mathscr{P}$ is a $m \times n$ disparity map and $\boldsymbol{p_{ij}}$ is a vertex (or node) at the site of (i, j) with a label of disparity d_{ij}. Because more candidates taken into consideration usually make the inference of a true disparity intractable, only the neighbours adjacent to $\boldsymbol{p_{ij}}$ are considered for stereo matching [16]. This is also known as a pairwise MRF. In general, $k = 4$ and $\mathscr{N}$ is a four-connected neighbourhood system. $\mathscr{E}_1 = (\boldsymbol{p_{ij}}, \boldsymbol{n_{1p_{ij}}})$, $\mathscr{E}_2 = (\boldsymbol{p_{ij}}, \boldsymbol{n_{2p_{ij}}})$, $\mathscr{E}_3 = (\boldsymbol{p_{ij}}, \boldsymbol{n_{3p_{ij}}})$ and $\mathscr{E}_4 = (\boldsymbol{p_{ij}}, \boldsymbol{n_{4p_{ij}}})$ are adjacent edges sharing the vertex $\boldsymbol{p_{ij}}$. The disparity of $\boldsymbol{p_{ij}}$ tends to have a strong correlation with its vicinities, while it is linked implicitly to any other random nodes in the disparity map. In [16], the joint probability of

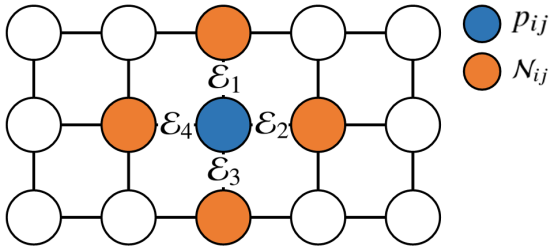


Figure 2.8: Markov random fields.

the MRF is written as:

$$P(\boldsymbol{p}, q) = \prod_{\boldsymbol{p_{ij}} \in \mathscr{P}} \Phi(\boldsymbol{p_{ij}}, q_{\boldsymbol{p_{ij}}}) \prod_{\boldsymbol{n}_{\boldsymbol{p_{ij}}} \in \mathscr{N}_{ij}} \Psi(\boldsymbol{p_{ij}}, \boldsymbol{n}_{\boldsymbol{p_{ij}}}) \tag{2.41}$$

where $q_{\boldsymbol{p_{ij}}}$ represents the intensity differences, $\Phi(\cdot)$ expresses the compatibility between possible disparities and the corresponding intensity differences, and $\Psi(\cdot)$ expresses the compatibility between $\boldsymbol{p_{ij}}$ and its neighbourhood system. Now, the aim of finding the best disparity is equivalent to maximising the probability in Eq. 2.41. This can be realised by formulating Eq. 2.41 as an energy function:

$$E(\boldsymbol{p}) = \sum_{\boldsymbol{p_{ij}} \in \mathscr{P}} D(\boldsymbol{p_{ij}}, q_{\boldsymbol{p_{ij}}}) + \sum_{\boldsymbol{n}_{\boldsymbol{p_{ij}}} \in \mathscr{N}_{ij}} V(\boldsymbol{p_{ij}}, \boldsymbol{n}_{\boldsymbol{p_{ij}}}) \tag{2.42}$$

$D(\cdot)$ and $V(\cdot)$ are two energy functions. $D(\cdot)$ corresponds to the matching cost and $V(\cdot)$ determines the aggregation from the neighbours. In the MRF model, the method to formulate an adaptive $V(\cdot)$ is important because the intensity in discontinuous areas usually varies greatly from that of its neighbours [58]. However, the process of minimising the energy function in Eq. 2.42 results in high computational complexities, making real-time performance challenging.

2.3.3.4 Disparity Refinement

The process of disparity refinement usually involves several post-processing steps, such as the LRC check, subpixel enhancement and weighted median filtering [32]. The LRC check removes most of the occluded areas that are only visible in one image and provides an outlier in the disparity map [11]. Furthermore, for some applications which require millimetre accuracy in 3-D reconstruction,

a disparity error larger than one pixel may result in a non-negligible difference in the reconstruction result [40]. Therefore, subpixel enhancement provides an easy way to increase the resolution of the disparity map by simply interpolating the matching costs around the initial disparity [32]. Moreover, a median filter is always applied to the disparity map to fill the holes and remove the incorrect matches [32]. However, the above disparity refinement algorithms are not always necessary and the sequential use of these steps depends entirely on the target application.

2.3.3.5 Algorithm Evaluation Methods

In computer stereo vision, speed and accuracy are two key aspects and they are always pitted against each other [40]. Therefore, the performance evaluation of a disparity estimation algorithm usually involves both of these two aspects. The overall performance of a proposed algorithm entirely depends on the trade-off between speed and precision [13].

As for the evaluation in terms of accuracy, Barron et al. proposed two general approaches to compute the error statistics of an estimated disparity map ℓ^{et} with respect to the ground truth data ℓ^{gt} [59]. The functions of these two error computing methods are shown in Eq. 2.43 and Eq. 2.44, where N_d represents the total number of disparities used for evaluation.

- Root-Mean-Squared (RMS) error e_{RMS}:

$$e_{RMS} = \sqrt{\frac{1}{N_d} \sum_{(u,v)} |\ell^{et}(u,v) - \ell^{gt}(u,v)|^2} \tag{2.43}$$

- Percentage of Error Pixels e_{PEP} (threshold: δ_d pixels):

$$e_{PEP} = \frac{1}{N_d} \sum_{(u,v)} \Big(|\ell^{et}(u,v) - \ell^{gt}(u,v)| > \delta_d \Big) \tag{2.44}$$

Furthermore, Scharstein and Szeliski also proposed to compute the error statistics in three different types of regions: texture-less, occluded, discontinuity [32, 33]. The corresponding percentages of error pixels are denoted as $e^{\mathscr{T}}_{PEP}$,

$e^{\mathscr{O}}_{PEP}$ and $e^{\mathscr{D}}_{PEP}$. By comparing the average percentages of error pixels in different regions, the accuracy performance of a disparity estimation algorithm can be evaluated more comprehensively.

In addition to accuracy, the execution speed of a chosen disparity estimation algorithm is also quantified in order to provide a complete evaluation of the overall performance. However, due to the fact that image size and disparity range are not constant among different datasets, a general way to depict the runtime performance is given in millions of disparity evaluations per second Mde/s [13] as follows:

$$Mde/s = \frac{u_{max} v_{max} d_{max}}{t} 10^{-6} \tag{2.45}$$

However, the runtime of a chosen disparity estimation algorithm greatly varies on different platforms. Therefore, some authors mainly focus on the implementation of some complex stereo matching algorithms to achieve a real-time performance, and the processing speed of these algorithms can be greatly boosted by exploiting the parallel computing architecture [11].

2.4 Lane Detection

The state-of-the-art lane detection algorithms can be grouped into two main categories: feature-based and model-based [60]. The feature-based algorithms extract the local, meaningful and detectable parts of an image, such as edges, texture and

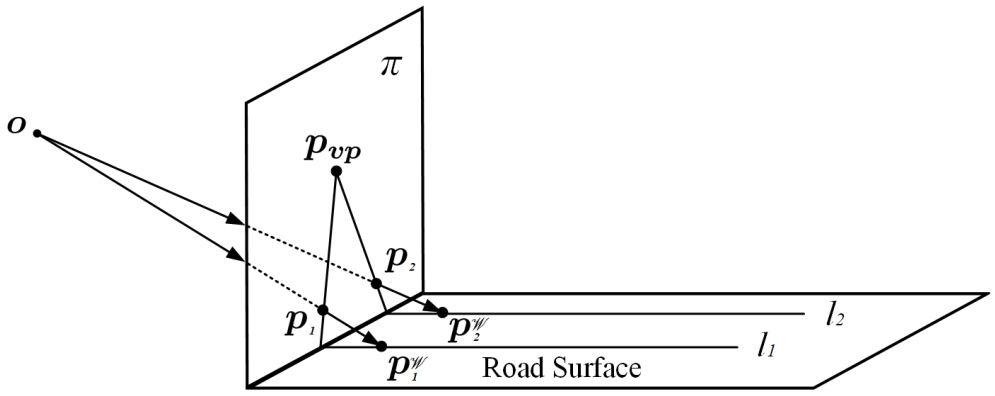


Figure 2.9: Vanishing point

colour, to segment lanes and road boundaries from the background [61]. On the other hand, the model-based algorithms try to represent the lanes with a mathematical equation based upon some common road geometry assumptions [62]. The most commonly used lane models include: linear, parabolic, linear-parabolic and spline. The linear model works well for the lanes with a low curvature, as demonstrated in [42, 63]. However, a more flexible road model is inevitable when the lanes with a higher curvature exist. Therefore, some algorithms [64–67] use a parabolic model to represent the lanes with a constant curvature. For some more complex cases, Jung et al. proposed a linear-parabolic combined lane model, where the nearby lanes are represented as linear models, whereas the far ones are modelled as parabolas [68]. In addition to the models mentioned above, the spline model is an alternative way to interpolate the lane pixels into an arbitrary shape [62, 69]. However, the more parameters introduced into a flexible model, the higher will be the computational complexity of the algorithm. Therefore, some authors turn their focus on some additional important properties of 3-D imaging techniques instead of being limited to only 2-D information.

One of the most prevalently used methods is Inverse Perspective Mapping (IPM). With the assumption that two lanes are parallel to each other in the WCS, IPM is able to map a 3-D scenery into a 2-D bird's eye view [70]. Furthermore, many researchers [6, 71–74] proposed to use the vanishing point $\boldsymbol{p_{vp}} = [u_{vp}, v_{vp}]^\top$ to model lane markings and road boundaries, where u_{vp} and v_{vp} represent the vertical and horizontal coordinates of the vanishing point, respectively. An example of vanishing point is illustrated in Figure 2.9, where l_1 and l_2 are two straight lines which are parallel to each other. Two 3-D points $\boldsymbol{p_1}^{\mathscr{W}}$ and $\boldsymbol{p_2}^{\mathscr{W}}$ in the WCS are projected on the image plane π and their corresponding points in the ICS are $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$, respectively. Therefore, the projections of l_1 and l_2 in the image are two straight lines and they intersect at the vanishing point $\boldsymbol{p_{vp}}$. However, their algorithms work well only if the road surface is assumed to be flat or the camera parameters are known. Therefore, some researchers pay closer attention to the disparity information which can be provided by either active sensors, e.g., radar and laser, or passive sensors, e.g., stereo cameras [6]. Since Labayrade et al. proposed the concept of "v-disparity" [75], disparity information has been widely used to enhance the robustness of the lane detection systems.

The work presented in [6] shows a particular instance where the disparity information is successfully combined with a lane detection algorithm to estimate $\boldsymbol{p_{vp}}$ for a non-flat road surface. At the same time, the obstacles contain a lot of redundant information which can be eliminated by comparing the actual and fitted disparity values. However, the estimation of $\boldsymbol{p_{vp}}$ suffers from the outliers when performing the Least Squares Fitting (LSF), and the lanes are sometimes unsuccessfully detected because the selection of plus-minus peaks is not always effective. Moreover, achieving real-time performance is still a challenging task in [6] because of the intensive computational complexity of the algorithm.

2.5 Road Surface 3-D Reconstruction

3-D reconstruction methods can be classified as laser scanner-based, Microsoft Kinect-based and passive sensor-based. The laser scanner (see Figure 2.10a) col-

(a)

(b)

(c)

Figure 2.10: 3-D reconstruction equipments. (a) laser scanner [3]. (b) Microsoft Kinect [4]. (c) ZED stereo camera.

lects the reflected laser pulse from an object to construct its accurate 3-D model [76]. Although it provides accurate modelling results, the laser scanner equipment used for road condition analysis is still costly [77]. As for the methods based on the Microsoft Kinect sensor (see Figure 2.10b), the depth measurement for the outdoor environment is somewhat ineffective, especially for materials which strongly absorb the infrared light [78]. Therefore, the passive sensor-based methods, e.g., stereo vision (see Figure 2.10), are more capable of reconstructing the 3-D road surface for condition assessment or damage detection.

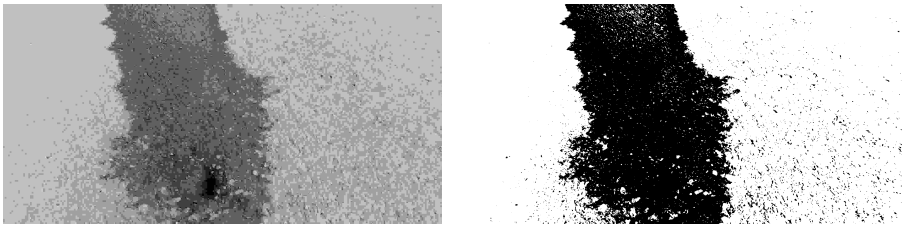
To reconstruct a real-world environment with passive sensing techniques, multiple camera views are required [8]. Images from different viewpoints can be captured using either a single moveable camera or an array of cameras [9]. If the stereo rig is well-calibrated, the main work of 3-D reconstruction turns out to be disparity estimation.

2.6 Pothole Detection

The vision-based pothole detection techniques can be grouped into two main categories: 2-D image processing-based and 3-D modelling-based. The state-of-the-art algorithms for these two categories will be discussed in subsections 2.6.1 and 2.6.2, respectively.

2.6.1 2-D Image Processing-Based Pothole Detection Algorithms

The pothole detection algorithms based on 2-D image processing usually consist of three steps: image segmentation, shape extraction and object recognition [77]. Firstly, the gray-scale images are segmented using some histogram-based thresholding algorithms, e.g., triangle [79] and Otsu's [80]. Compared to the triangle thresholding utilised in [79], Otsu's thresholding which minimises the intra-class variance, shows a better performance in terms of distinguishing a distress region from the background [80]. Before extracting the region of a potential pothole, the segmented images are processed with some commonly used image processing algorithms, such as filtering, morphological operation and region growing, to re-



(a) (b)

Figure 2.11: Example of failed segmentation of distress and non-distress areas for a gray-scale image. (a) gray-scale image. (b) segmentation result. The distress and non-distress areas in (b) are shown in black and white colours, respectively.

duce the redundant information and clarify the outline of the target region [81], [82]. Then, the extracted region around a potential pothole is interpolated into an elliptical shape [79]. The texture inside the ellipse is compared with the texture of the non-distress region. If the former is grainier and coarser than the latter, the ellipse shape extracted in the second step is designated as a pothole [81].

However, the segmentation of a gray-scale image is always affected by the actual environmental conditions. An example of a failed segmentation is depicted in Figure 2.11, where the water on the road surface makes the segmentation ambiguous. Therefore, some authors proposed to carry out the segmentation algorithm on the depth image, and thus the distress and non-distress areas can be separated more accurately [83, 84]. Furthermore, the shapes of actual potholes are always irregular, making the geometric and textural assumptions infeasible. But on the other hand, 3-D modelling-based algorithms are capable of detecting irregular shaped potholes. Moreover, the spatial structure of a detected pothole cannot be illustrated explicitly in 2-D images [40], and therefore 3-D information is required to measure pothole volumes. In general, 3-D modelling-based pothole detection algorithms are more than capable of overcoming the disadvantages mentioned above.

2.6.2 3-D Modelling-Based Pothole Detection Algorithm

The 3-D information used for pothole detection is mainly provided by: laser scanner [83], Microsoft Kinect [84] and passive sensor [85–89]. Among them, the laser scanning equipment mounted on Digital Inspection Vehicles (DIVs) is still

cost-intensive for road condition assessment [77], and the Microsoft Kinect sensors suffer from infrared saturation in direct sunlight in an outdoor environment [90]. Therefore, the passive sensors are more suitable for acquiring the 3-D road surface for pothole detection.

When using passive sensors to reconstruct the road sceneries, multiple camera views are required [8]. These can be obtained by using either a single movable camera or multiple synchronised cameras. For example, Zhang and Elaksher mounted a single camera on an Unmanned Aerial Vehicle (UAV) to reconstruct the pavements for pothole detection [85]. In addition, stereo vision-based pothole detection systems have also been developed since Barsi et al. first applied a binocular system to detect potholes [86]. In [87], the 3-D point cloud acquired using a stereo system is fitted to a quadratic surface using the LSF. The potholes are identified by comparing the difference between the observed and interpolated road surface. Different potholes are also labelled using Connected Component Labelling (CCL). Ozgunalp et al. improved on the surface modelling method by integrating the surface normal into the procedure of the LSF [88]. In [89], the authors proposed to perform the surface fitting in the disparity domain instead of the Euclidean domain, and the LSF is performed together with the Random Sample Consensus (RANSAC). The work in [40] proposed an efficient road surface 3-D reconstruction algorithm based on stereo vision technology, which can provide highly accurate disparity maps and 3-D modelling results.

2.7 Heterogeneous System

2.7.1 Multi-Threading CPU

In order to improve the execution speed, Open Multi-Processing (OpenMP) can be used to break a serial code into independent chunks for parallel processing [11]. OpenMP consists of three main components: work sharing, data sharing and synchronisation. Work sharing specifies which part of the serial code is going to be parallelised, data sharing specifies an appropriate scheduling model, and synchronisation determines how data is shared [91]. The process of OpenMP can be depicted using a fork-join model as shown in Figure 2.12.

2.7.2 GPU

Graphics processors have been widely used in 3-D computer vision to accelerate some computationally intensive but parallelly efficient algorithms for a real-time purpose. The general architecture of the graphics processors is shown in Figure 2.13. Compared with a Central Processing Unit (CPU) which consists of a low number of cores optimised for sequentially serial processing, the Graphics Processing Unit (GPU) has a highly parallel architecture which is composed of hundreds or thousands of lighter cores to handle multiple tasks concurrently.

As shown in Figure 2.13, a GPU consists of N Streaming Multi-Processors (SMs) with M Streaming Processors (SPs) on each of them. The Single Instruction Multiple Data (SIMD) architecture allows the SPs on the same SM to execute the same instruction but process different data at each clock cycle [92]. The device has its own Dynamic Random Access Memory (DRAM) which consists of global memory, constant memory and texture memory. The latter can communicate with the host memory via the Graphical/Memory Controller Hub (GMCH) and the I/O Controller Hub (ICH) which are also known as the Intel northbridge and the Intel southbridge, respectively. Each SM has four types of on-chip memories: register, shared memory, constant cache and texture cache. Since they are on-chip memories, the constant cache and texture cache are utilised to speed up the data fetching from the constant memory and texture memory, respectively. Due to the fact that the shared memory is small, it is used for the duration of processing a block. The register is only visible to the thread. The details of

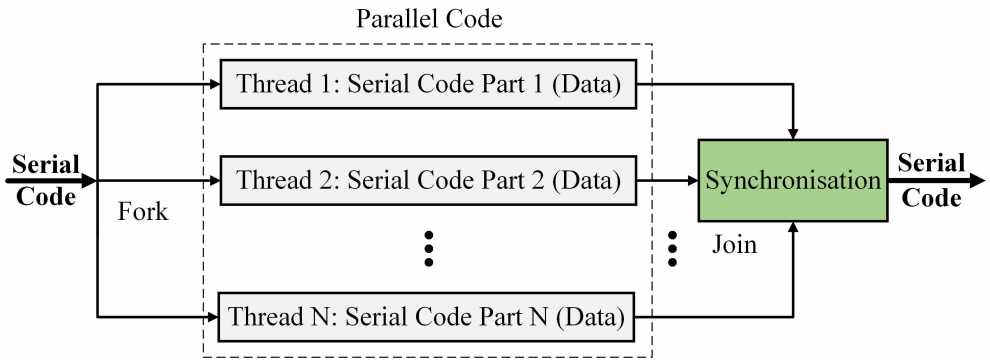


Figure 2.12: OpenMP

different types of memories are illustrated in Table 2.7.2.

Table 2.1: GPU memory architecture.

Memory	Location	Cached	Access	Scope
register	on-chip	n/a	r/w	one thread
shared	on-chip	n/a	r/w	all threads in a block
global	off-chip	no	r/w	all threads + host
constant	off-chip	yes	r	all threads + host
texture	off-chip	yes	r	all threads + host

In CUDA C programming, the threads are grouped into a set of 3-D thread blocks which are then organised as a 3-D grid. The kernels are defined on the host using the CUDA C programming language. Then, the host issues the commands that submit the kernels to devices for execution [93]. Only one kernel can be executed at a given time. Once a thread block is distributed to an SM, the threads are divided into groups of 32 parallel threads which are executed by SPs.

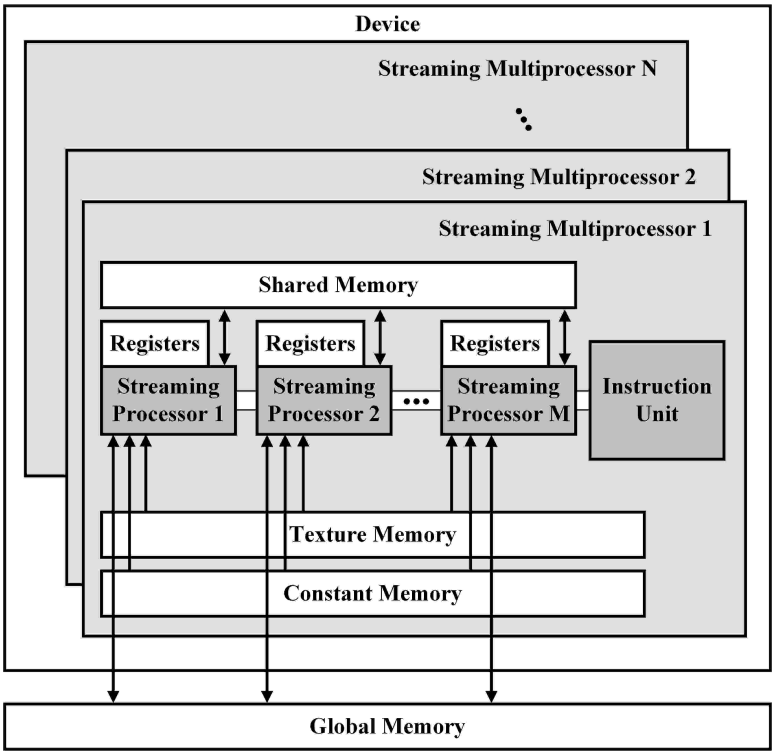


Figure 2.13: Brief overview of general GPU architecture.

Each group of 32 parallel threads is known as a warp [92]. Therefore, the size of a thread block is usually chosen as a multiple of 32 to ensure efficient data processing.

Chapter 3

Real-Time Disparity Map Estimation System Based on Optimised Normalised Cross-Correlation and Propagated Search Range

In recent years, computer stereo vision technique has been prevalently used in various prototype vehicle road tests to provide the depth information for autonomous vehicles. This greatly helps to enhance the robustness of various subsystems, e.g., lane detection and obstacle detection, in the Advanced Driver Assistance Systems (ADAS). In this chapter, an efficient stereo matching algorithm based on ground plane (GP) assumption is presented to acquire dense disparity maps ℓ for automotive applications. The estimated disparity maps are then utilised in Chapter 4 to improve the process of dense vanishing point estimation. The proposed algorithm is developed from [28], where the search range at the position of (u, v) is propagated from three estimated neighbouring disparities $\ell(u-1, v+1)$, $\ell(u, v+1)$ and $\ell(u+1, v+1)$. Taking the advantage of search range propagation (SRP), only a small portion of disparity space needs to be taken into account, which greatly reduces the expensive computations. In order to further improve

the execution speed of the algorithm, the standard NCC cost function is factorised into five independent parts. The computations of μ_l, μ_r, σ_l and σ_r are accelerated using four integral images I_l, I_r, I_{l^2} and I_{r^2} and their values are stored in a static program storage for direct indexing. The parallel computing architectures, i.e., OpenMP and CUDA, are also exploited for the real-time purpose. The experimental results illustrate that the proposed stereo matching algorithm is capable of providing dense disparity information with a low percentage of error pixels and the implementation on GPU performs in real time. The main contributions of this Chapter are published in [11] and [94].

The remainder of this chapter is organised as follows: Section 3.1 gives an overview of the proposed system. Section 3.2 describes the proposed disparity estimation algorithm. Section 3.3 discusses the implementations on a multi-threading CPU and a GPU. Section 3.4 illustrates the experimental results and Section 3.5 concludes the chapter.

3.1 System Overview

As discussed in Section 2.3, matching speed and disparity accuracy are two key aspects of computer stereo vision. Although the global algorithms can provide some accurate disparity maps by solving the stereo matching problem using some sophisticated optimisation techniques, e.g., GC and BP, it is very challenging for them to achieve real-time performance without specialised hardware accelerators [13]. On the other hand, some efficient local algorithms, e.g., SRP, GCS and DTSM, are developed based on Search Range Constraints (SRC). These algorithms not only greatly reduce the mismatches caused by ambiguities but also ensure a good performance in terms of speed. To further boost the processing speed and at the same time minimise expensive computations, the cost functions can be simplified. For example, Lin et al. proposed an optimisation methodology for the computation of the NCC by dividing the standard equation into four independent parts and accelerating their computations using sliding windows (SW) [95]. Compared with the SW, integral image is a more efficient algorithm in terms of computing μ and σ. Using an integral image, the sum of pixel intensities over a rectangular region of the image can be calculated with only four operations [96].

Therefore, the proposed system is developed from the algorithm in presented [28] and the process of stereo matching is further accelerated using integral images.

The proposed algorithm in this chapter consists of three main steps: μ and σ memorisation, disparity estimation for row v_{max} with a full search range, and disparity estimation for the rest of image with a propagated search range. The incorrect disparities in the occluded areas are also removed using the LRC check. To yield a real-time performance, the proposed algorithm is implemented on an eight-thread CPU using OpenMP and a state-of-the-art GPU using CUDA.

3.2 Algorithm Description

3.2.1 Memorisation

In this chapter, the input stereo image pairs are assumed to be well-rectified. Due to the insensitivity to the intensity difference, the NCC is utilised as the cost function to measure the similarity between two blocks, as shown in Eq. 2.37. Each block chosen from the left image (see Figure 3.4a) is matched with a series of blocks on the same epipolar line in the right image (see Figure 3.4b). The block pair with the highest correlation cost is then selected as the best correspondence, and the shifting distance between them is the desirable disparity d.

However, when the left block is selected, the computations of μ_l and σ_l are always repeated because d is only used to select the positions of the right blocks

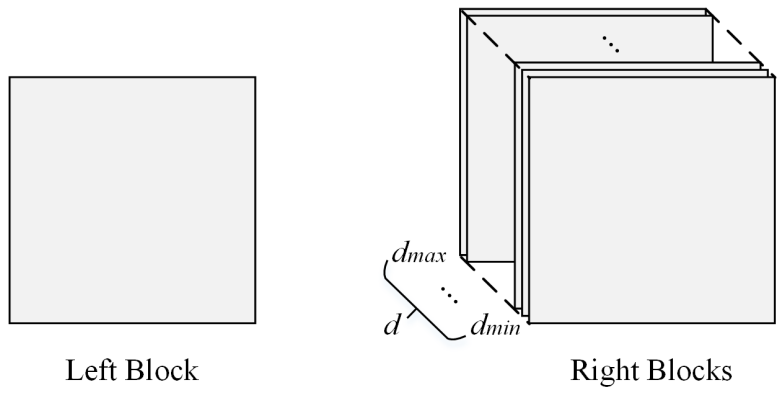


Figure 3.1: Block matching.

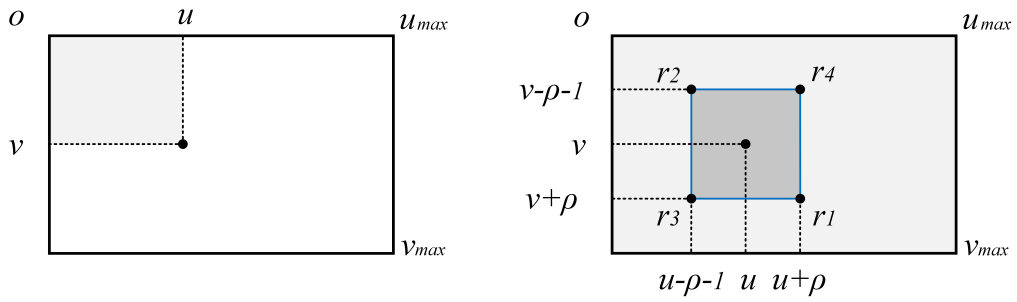


Figure 3.2: Integral image processing. (a) original image. (b) integral image.

for stereo matching, as shown in Figure 3.1. The computations of μ_r and σ_r are also repeated because each right block needs to be compared with a set of d left blocks on the same epipolar line. Therefore, the four independent parts μ_l, μ_r, σ_l and σ_r can be pre-calculated and stored in a static program storage for direct indexing. The integral images can be used to compute μ_l and μ_r efficiently [97], which is illustrated in Figure 3.2. The integral image algorithm consists of two steps: integral image initialisation and value indexing from the initialised reference. In the first step, for a discrete image i whose pixel intensity at (u, v) is $i(u, v)$, its integral image intensity $I(u, v)$ at the position of (u, v) is defined as:

$$I(u,v) = \sum_{x \le u,\ y \le v} i(x,y) \tag{3.1}$$

Algorithm 1 details the implementation of the integral image initialisation, where I is calculated serially based on its previous neighbouring results to minimise unnecessary computations.

After initialising an integral image, the sum $s(u, v)$ of pixel intensities within a square block whose side length is $2\rho+1$ and centre is (u, v) can be computed using four references $r_1 = I(u+\rho, v+\rho)$, $r_2 = I(u-\rho-1, v-\rho-1)$, $r_3 = I(u-\rho-1, v+\rho)$ and $r_4 = I(u+\rho, v-\rho-1)$ as follows:

$$s(u,v) = r_1 + r_2 - r_3 - r_4 \tag{3.2}$$

The mean $\mu(u, v) = s(u, v)/n$ of the intensities within the selected block is

Algorithm 1: Integral image initialisation

Input : original image: i
Output: integral image: I

$I(u_{min}, v_{min}) \leftarrow i(u_{min}, v_{min})$;
for $u \leftarrow u_{min} + 1$ *to* u_{max} **do**
  $I(u, v_{min}) \leftarrow I(u-1, v_{min}) + i(u, v_{min})$;
end
for $v \leftarrow v_{min} + 1$ *to* v_{max} **do**
  $I(u_{min}, v) \leftarrow I(u_{min}, v-1) + i(u_{min}, v)$;
end
for $u \leftarrow u_{min} + 1$ *to* u_{max} **do**
  for $v \leftarrow v_{min} + 1$ *to* v_{max} **do**
    $I(u, v) \leftarrow I(u, v-1) + I(u-1, v)$
    $-I(u-1, v-1) + i(u, v)$;
  end
end

then stored in a static program storage for the computations of σ and c. To simplify the computations of σ_l and σ_r, we rearrange Eq. 2.38 and Eq. 2.39 as shown in Eq. 3.3 and Eq. 3.4, respectively.

$$\sigma_l = \sqrt{\sum_{x=u-\rho}^{x=u+\rho} \sum_{y=v-\rho}^{y=v+\rho} {i_l}^2(x, y)/n - {\mu_l}^2} \tag{3.3}$$

$$\sigma_r = \sqrt{\sum_{x=u-\rho}^{x=u+\rho} \sum_{y=v-\rho}^{y=v+\rho} {i_r}^2(x-d, y)/n - {\mu_r}^2} \tag{3.4}$$

where $\sum {i_l}^2$ and $\sum {i_r}^2$ are dot products. Similarly, the computations of $\sum {i_l}^2$ and $\sum {i_r}^2$ can be accelerated by initialising two integral images I_{l^2} and I_{r^2} as the references for value indexing. The expressions of I_{l^2} and I_{r^2} are as follows:

$$\begin{aligned} I_{l^2}(u, v) &= \sum_{x \le u,\ y \le v} {i_l}^2(x, y) \\ I_{r^2}(u, v) &= \sum_{x \le u,\ y \le v} {i_r}^2(x, y) \end{aligned} \tag{3.5}$$

Therefore, the standard deviations σ_l and σ_r can also be calculated and stored in a static program storage for the efficient computation of c_{NCC} as follows:

$$c_{NCC}(u,v,d) = \frac{1}{n\sigma_l\sigma_r}\left[\sum_{x=u-\rho}^{x=u+\rho}\sum_{y=v-\rho}^{y=v+\rho} i_l(x,y)i_r(x-d,y) - n\mu_l\mu_r\right] \tag{3.6}$$

According to Eq. 3.6, only $\sum i_l i_r$ needs to be calculated during the stereo matching. Hence, with the values of μ_l, μ_r, σ_l and σ_r able to be indexed directly, Eq. 2.37 is simplified as a dot product. The performance improvement achieved by factorising the NCC equation will be discussed Section 3.4.

3.2.2 Search Range Propagation

In this paper, the disparities are estimated iteratively row by row from row v_{max} to row v_{min}. In the first iteration, the stereo matching goes for a full search range $SR = \{sr | sr \in [d_{min}, d_{max}]\}$. Then, the search range for stereo matching at the position of (u,v) is propagated from three estimated neighbouring disparities $\ell(u-1,v+1)$, $\ell(u,v+1)$ and $\ell(u+1,v+1)$ using Eq. 3.7 [11], where τ is the bound of the search range and it is set to 1 in the proposed system. The estimated left and right disparity maps, i.e., ℓ^{lf} and ℓ^{rt}, are illustrated in Figure 3.4c and Figure 3.4d, respectively. More details on the SRP-based disparity estimation are given in algorithm 2. The performance of the SRP-based stereo will be discussed in Section 3.4.

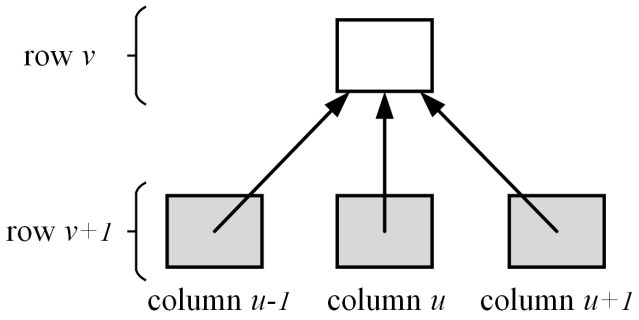


Figure 3.3: Search range propagation.

$$SR = \bigcup_{k=u-1}^{u+1} \{sr | sr \in [\ell(k, v+1) - \tau, \ell(k, v+1) + \tau]\} \tag{3.7}$$

Algorithm 2: SRP-based disparity map estimation

Input : left image, right image;
left mean map, right mean map;
left standard deviation map, right standard deviation map;
Output: disparity map

```
estimate the disparities for row v_max;
for v ← v_max − 1 to v_min do
    for u ← u_min to u_max do
        propagate the search range from row v + 1 using Eq. 3.7;
        estimate the disparity for (u, v);
    end
end
```

3.2.3 Left-Right Consistency Check

For various disparity map estimation algorithms, the pixels that are only visible in one disparity map are a major source of the matching errors. Due to the uniqueness constraint of the correspondence, for an arbitrary pixel (u, v) in the left disparity map ℓ^{lf}, there exists at most one correspondence in the right disparity map ℓ^{rt}, namely [18]:

$$\ell^{lf}(u, v) = \ell^{rt}(u - \ell^{lf}(u, v), v) \tag{3.8}$$

Pixels that are only visible in one disparity map are marked as uncertainties. A left-right consistency check is performed to remove these half-occluded areas. Although the LRC check doubles the computational complexity by re-projecting the computed disparity values from one image to the other one, most of the incorrect half-occluded pixels can be removed and an outlier can be found. For the estimation of ℓ^{rt}, the memorisation of μ_l, μ_r, σ_l and σ_r is unnecessary because they have already been calculated when estimating ℓ^{lf}. More details on the LRC check is given in algorithm 3, where tr_{LRC} is the threshold and it is set to 3 in

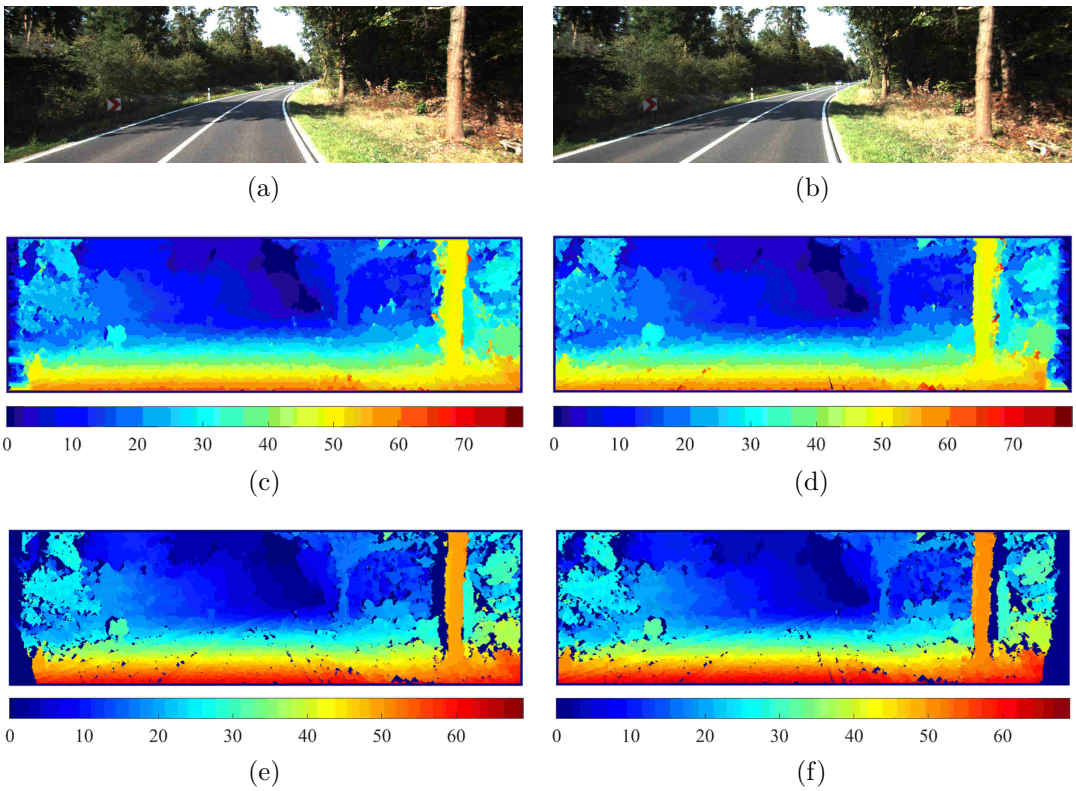


Figure 3.4: Disparity map estimation. (a) left image. (b) right image. (c) left disparity map ℓ^{lf}. (d) right disparity map ℓ^{rt}. (e) left disparity map processed with the LRC check. (f) right disparity map processed with the LRC check.

Algorithm 3: LRC check

Input : left disparity map: ℓ^{lf}
right disparity map: ℓ^{rt}
Output: disparity map: ℓ

```
for v ← v_min to v_max do
    for u ← u_min to u_max do
        if abs(ℓ^lf(u,v) − ℓ^rt(u − ℓ^lf(u,v), v)) > tr_LRC then
            ℓ(u,v) ← 0;
        else
            ℓ(u,v) ← ℓ^lf(u,v);
        end
    end
end
```

this chapter. The corresponding LRC check results are shown in Figure 3.4e and Figure 3.4f.

3.3 Implementations

The main purpose of this chapter is to improve the processing speed of the algorithm proposed in [28]. To achieve a real-time performance, the proposed algorithm is implemented on an Intel Core i7-4720HQ CPU and an NVIDIA GTX 970M GPU using OpenMP and CUDA, respectively. The implementation procedures are details in this section.

3.3.1 CPU Implementation

This subsection presents the details on the CPU implementation using eight threads. The performance results with different number of threads will be discussed in Section 3.4.

The integral images are first initialised serially using one thread. Then, the for loops in memorisation stage are divided among eight threads using *omp for* clause. *dynamic* is selected as the scheduling model because it performs better in terms of distributing unequal subtasks to each thread. When a thread finishes the execution of a chunk of data, it retrieves the next chunk. Meanwhile, μ_l, μ_r, σ_l and σ_r are declared as *private* variables to make each thread have its own copy. As for the synchronisation, *nowait* clause is utilised to ignore the implicit barrier of *for* pragma. The rest of the algorithm is parallelised using *omp sections*, and the serial code is equally divided into eight sub-blocks to execute concurrently. Finally, the result from each thread is combined to obtain the disparity map.

3.3.2 GPU Implementation

Graphics cards have been widely used to accelerate the processing of various 3-D computer vision algorithms which are computationally intensive but can be executed efficiently in parallel. In the GPU architecture, a thread is more likely to fetch the memory from the closest addresses that its nearby threads accessed, which makes the use of cache impossible [92]. Therefore, the author utilises the

texture memory which is read-only and cached on-chip to optimise the caching for 2-D spatial locality during the stereo matching. Firstly, two 2-D texture objects are created for the left and right images. Then, the texture objects are bound directly to the address of the global memory. The value of a pixel $i_l(x, y)$ or $i_r(x, y)$ is then fetched from the texture memory to reduce the memory requests from the global memory. In the memorisation stage, the images are divided into a group of 32×8 thread blocks, and each of them is divided into eight warps which are then processed in parallel by a set of SPs. Then the disparities on row v_{max} are estimated in parallel using a set of 64×1 thread blocks. The image intensities i_l and i_r are also fetched from the texture references for the computation of $\sum i_l i_r$, whereas the values of μ_l, μ_r, σ_l and σ_r are indexed directly from the global memory to reduce the unnecessary computations. As for the disparity estimation for the rest of the disparity map, the search range at the position of (u, v) is independent to the horizontal neighbouring disparities at the position of

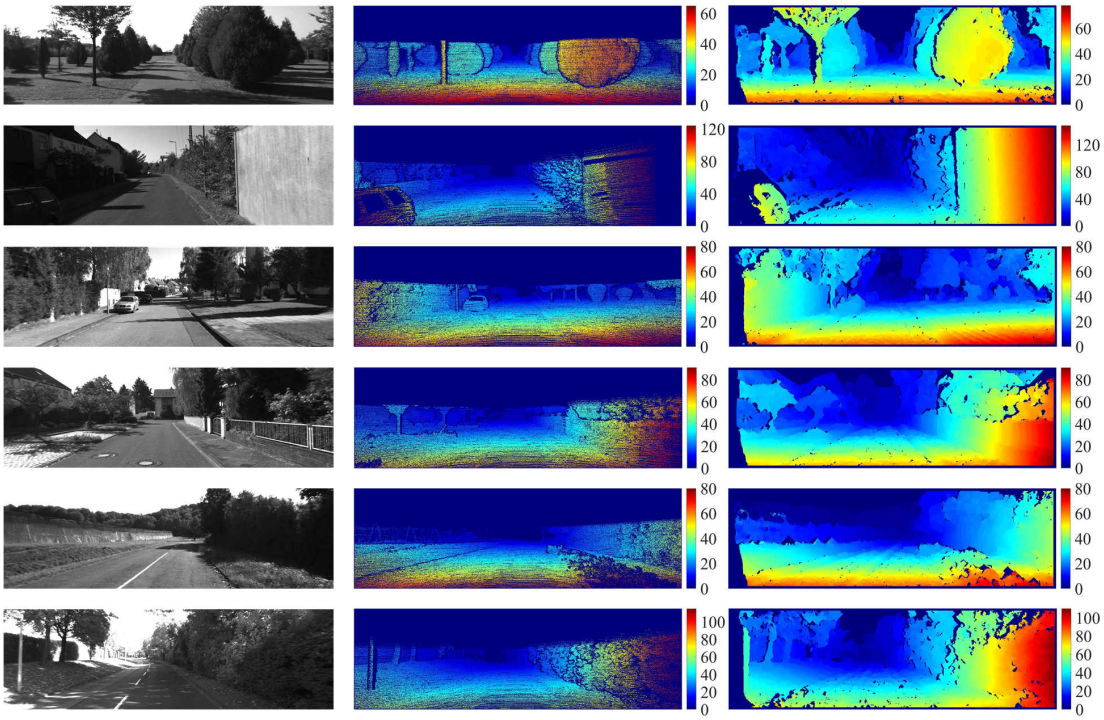


Figure 3.5: Experimental results of KITTI stereo 2012 dataset [5]. ρ and τ are set to 5 and 1, respectively.

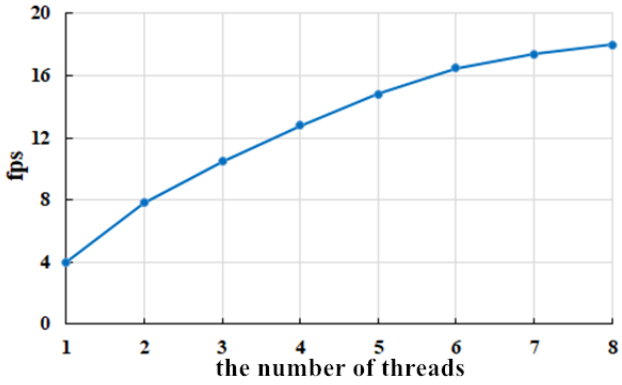


Figure 3.6: Processing speed with respect to different number of threads.

$(u-1, v)$ and $(u+1, v)$, and only relies on the previously estimated disparities located on row $v+1$. Therefore, the proposed algorithm is performed iteratively from row v_{max} to row v_{min}, and each row is processed in parallel using the strategy described in algorithm 2.

3.4 Experimental Results

The proposed disparity estimation algorithm is evaluated using the KITTI stereo 2012 dataset [5]. Some examples of the experimental results are shown in Figure 3.5, where the first column illustrates the input left gray-scale images, the second column shows the ground truth disparity maps, and the third column depicts our experimental results. The overall percentage of the error pixels is approximately 6.82% (error threshold: two pixels), which is around half of the rate obtained when using the GCS.

The proposed disparity estimation algorithm is implemented in C language on an Intel Core i7-4720HQ CPU (frequency: 2.6 GHz, the number of cores: 4) and an NVIDIA GTX 970M GPU (memory clock: 2500 MHz, the number of cuda cores: 1280). Firstly, the CPU performance with different number of threads is evaluated, as shown in Figure 3.6. It can be seen that the execution speed using two threads is twice than that using a single thread. However, the performance improvement becomes less distinct with increasing number of threads.

Next, the performance with respect to different values of ρ and τ is discussed.

Table 3.1: Processing speed of the proposed algorithm.

Platform	CPU threads	τ	ρ	fps
CPU	8	1	3	18
CPU	1	1	3	4
GPU	N/A	1	3	37
GPU	N/A	1	5	32
GPU	N/A	2	5	24

Table 3.2: Comparison between the algorithm in [28] and the proposed algorithm in terms of processing speed (unit: fps).

Algorithm in [28]	Proposed algorithm	ρ
59	28	1
62	33	2
70	37	3
86	42	4
137	63	5

The runtime of the algorithm on different platforms using different parameters is shown in Table 3.1. It can be observed that the execution speed of the algorithm decreases when either ρ or τ increases. Compared with the performance of CPU implementation using a single thread, the implementation on GPU speeds up the algorithm execution by around nine times. A speed of 37 fps is achieved when ρ and τ are set to 3 and 1, respectively.

After the memorisation, the values of μ and σ can be accessed directly from a static program storage for stereo matching, which greatly boosts the algorithm execution. The performance improvement achieved by using memorisation is shown in Figure 3.7, where t_{NCC} and t_{NCCM} represent the runtime of the conventional NCC-based stereo and the runtime of NCC-based stereo optimised with memorisation. From Figure 3.7, it can be seen that the memorisation speeds up the algorithm execution by about two times.

Furthermore, the comparison between the algorithm in [28] and the proposed algorithm in terms of processing speed is provided in Table 3.2, where both of the two algorithms are run on the GTX 970M GPU.

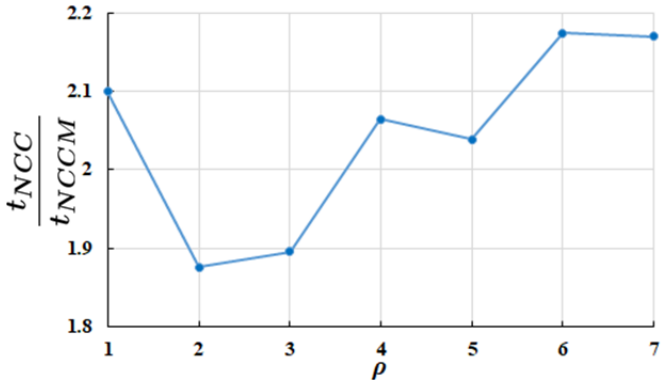


Figure 3.7: Runtime performance with respect to different values of ρ.

3.5 Conclusion

A real-time stereo vision system was presented in this chapter. To reduce the computational complexity, the standard NCC equation was factorised into five independent parts and their computations were accelerated using integral images. Furthermore, the search range at the position of (u, v) was propagated from three estimated neighbouring disparities $\ell(u-1, v+1)$, $\ell(u, v+1)$ and $\ell(u+1, v+1)$. This not only speeds up the algorithm execution but also reduces the ambiguities during the stereo matching. The incorrect disparities in the occluded areas were also removed by performing the LRC check, which further improves the accuracy of the estimated disparity maps. The proposed algorithm was implemented on a multi-threading CPU and a GPU using OpenMP and CUDA, respectively. The implementation on the GPU performs in real time.

Chapter 4

Real-Time Lane Detection System Based on Dense Vanishing Point Estimation

The detection of multiple curved lane markings on a non-flat road surface is still a challenging task for vehicular systems. To make an improvement, the depth information can be used to enhance the robustness of the lane detection systems. In this chapter, the proposed lane detection system is developed from [6] where the estimation of the dense vanishing point $\boldsymbol{p_{vp}} = [u_{vp}, v_{vp}]^\top$ is further improved using the disparity information. However, the outliers in the LSF severely affect the accuracy when estimating the vanishing point. Therefore, in this chapter the RANSAC is used to update the parameters of the road model iteratively until the percentage of the inliers exceeds a pre-set threshold. This significantly helps the system to overcome some suddenly changing conditions. Furthermore, the author proposes a novel lane position validation approach which computes the energy of each possible solution and selects all satisfying lane positions for visualisation. The proposed system is implemented on a heterogeneous system which consists of an Intel Core i7-4720HQ CPU and an NVIDIA GTX 970M GPU, and a processing speed of 143 fps has been achieved. Moreover, in order to evaluate the detection precision, 2495 frames including 5361 lanes are tested. It is shown that the overall successful detection rate is 99.5%. The main contributions

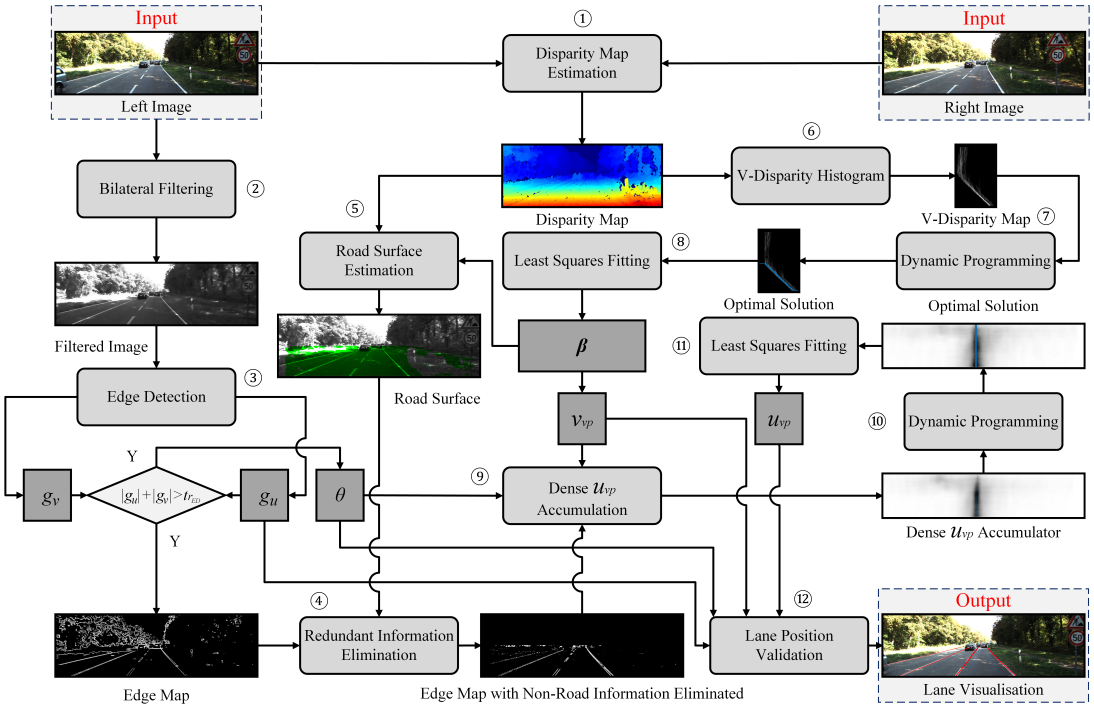


Figure 4.1: The block diagram of the proposed lane detection system.

of this chapter are published in [6] and [94].

4.1 System Overview

The proposed multiple lane detection system is composed of four main components: disparity map estimation, dense v_{vp} estimation, dense u_{vp} estimation and lane position validation. The block diagram of the proposed system is illustrated in Figure 4.1, where procedures 1 to 5 are processed on a GPU because they are more efficient for parallel processing but the serially-efficient procedures 6 to 12 are executed on a CPU.

Firstly, a disparity map is estimated using the algorithm described in Chapter 3. The disparity map is mainly used for:

- estimation of dense v_{vp},
- road surface estimation, and

- elimination of the redundant information.

In this chapter, the road surface is not assumed to be flat and the projection of the road disparities on the v-disparity map is modelled as a parabola. Compared with some quadratic pattern detectors, Dynamic Programming (DP) is a more efficient way to extract the path with the highest accumulations from the v-disparity map. The extracted path is then interpolated into a parabola using the LSF. However, the outliers in the LSF severely affect the accuracy of the vanishing point estimation. Therefore, the author proposes to update the parabola function iteratively using the RANSAC until the percentage of the inliers exceeds a preset threshold. This greatly improves the robustness of the proposed system. Since the bilateral filter performs better than the median filter in terms of edge preservation and noise elimination, it is utilised to reduce the unnecessary edges before estimating u_{vp}. v_{vp} and the orientation of each edge point in the road surface area are then used to estimate u_{vp}, where the RANSAC is employed to minimise the influence of the outliers on the LSF. An arbitrary lane marking or road boundary can thus be extracted using the vanishing point information. Finally, the author proposes a novel lane position validation approach which computes the energy of each possible solution and selects all satisfying lanes for visualisation.

The remainder of this chapter is structured as follows: Section 4.2 describes the proposed lane detection system. Section 4.3 evaluates the experimental results. Section 4.4 summarises the chapter.

4.2 System Description

4.2.1 Disparity Map Estimation

Since the stereo vision system presented in Chapter 3 greatly improves the trade-off between accuracy and speed, it is employed in this chapter to acquire the disparity information for the proposed lane detection system.

4.2.2 Dense v_{vp} Estimation

Since Labayrade et al. proposed the concept of "v-disparity" in 2002 [75], disparity information has been widely used to improve the detection of either obstacles or lanes. The v-disparity map is created by computing the histogram of each horizontal row of the disparity map. An example of the v-disparity map is shown in Figure 4.2a, which has two axes: disparity d and row number v. The value $m_v(d, v)$ represents the accumulation at the position of (d, v) in the v-disparity map. In [50], Hu et al. proved that the disparity projection of a flat road on the v-disparity map is a straight line: $d = f(v) = \alpha_0 + \alpha_1 v$. The parameter vector $\boldsymbol{\alpha} = [\alpha_0, \alpha_1]^\top$ can be obtained by using some linear pattern detectors, such as the Hough Transform (HT) [42, 98]. In this chapter, the disparity projection of a non-flat road surface on the v-disparity map is represented by a parabola model $d = f(v) = \beta_0 + \beta_1 v + \beta_2 v^2$. In this case, the DP is more efficient than some quadratic pattern detectors in terms of searching for every possible solution. The path with the highest accumulations can be extracted by minimising the energy in Eq. 4.1, where the term E_{data} penalises the solutions that are inconsistent with the observed data, E_{smooth} enforces the piecewise smoothness and λ is the smoothness parameter.

$$E = E_{data} + \lambda E_{smooth} \tag{4.1}$$

Eq. 4.1 is solved iteratively starting from $d = d_{max}$ and going to $d = 0$. In the first iteration, $E_{smooth} = 0$ and $E_{data} = -m_v(d_{max}, v)$. Then, E is computed based upon the previous iterations:

$$E(v)_d = -m_v(d, v) + \min_{\tau_v}[E(v - \tau_v)_{d+1} - \lambda_v \tau_v], \text{ s.t. } \tau_v \in [0, 6] \tag{4.2}$$

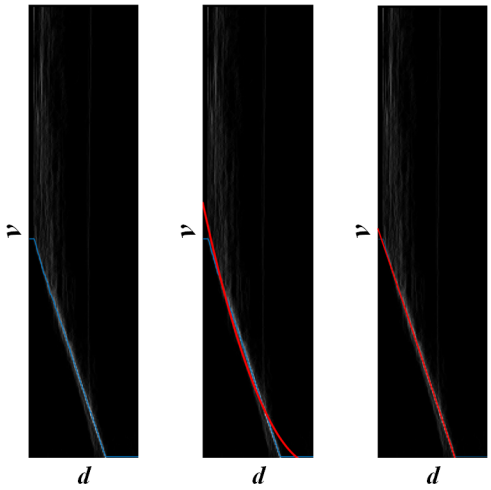
In each iteration, the index position of the minimum is saved into a buffer for the extraction of the desirable path. The buffer has the same size as the v-disparity map. The solution $\boldsymbol{M_v} = [\boldsymbol{d}, \boldsymbol{v}]^\top \in \mathbb{R}^{k \times 2}$ with the minimal energy is then selected as the optima, which is plotted in blue as shown in Figure 4.2.

The blue path includes k points. The two column vectors $\boldsymbol{v} = [v_0, v_1, \ldots, v_{k-1}]^\top$ and $\boldsymbol{d} = [d_0, d_1, \ldots, d_{k-1}]^\top$ record the row numbers and the disparity values, respectively. Therefore, the parameter vector $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2]^\top$ can be estimated by solving the least squares problem in Eq. 4.3. The parabola: $f(v) = \beta_0 + \beta_1 v + \beta_2 v^2$ is plotted in red, as shown in Figure 4.2b and Figure 4.2c.

$$\boldsymbol{\beta} = \arg\min_{\boldsymbol{\beta}} \sum_{j=0}^{k-1} (d_j - (\beta_0 + \beta_1 v_j + \beta_2 {v_j}^2))^2 \tag{4.3}$$

From Figure 4.2b, it can be observed that the outliers severely affect the accuracy of the LSF. To improve v_{vp} estimation, the RANSAC is utilised to update the inlier set $\mathscr{I}$ and the parameter vector $\boldsymbol{\beta}$ iteratively. This procedure is detailed in algorithm 4.

To determine whether a given candidate $[d_j, v_j]^\top$ belongs to $\mathscr{I}$, the corresponding squared residual $r_j = (d_j - f(v_j))^2$ needs to be computed. If r_j is smaller than a pre-set tolerance tr_v, the candidate is marked as an inlier and $\mathscr{I}$



(a) (b) (c)

Figure 4.2: DP and $\boldsymbol{\beta}$ estimation. (a) v-disparity map. (b) target solution obtained in [6]. (c) target solution obtained in the proposed system. The blue paths are the optimal solutions obtained using the DP. $f(v) = \beta_0 + \beta_1 v + \beta_2 v^2$ is plotted in red.

Algorithm 4: $\boldsymbol{\beta}$ estimation with the assistance of the RANSAC.

Input : optimal solution: $\boldsymbol{M_v} = [\boldsymbol{d}, \boldsymbol{v}]^\top$
Output: parameter vector: $\boldsymbol{\beta}$

```
do
    randomly select a specified number of candidates [d_j, v_j]^⊤;
    fit a parabola to the selected candidates and get β;
    determine the numbers of inliers and outliers: n_𝒥 and n_𝒪,
     respectively;
    remove the outliers from M_v;
while n_𝒥/(n_𝒥 + n_𝒪) < ε_v;
interpolate the candidates in the updated M_v into a parabola and get β;
```

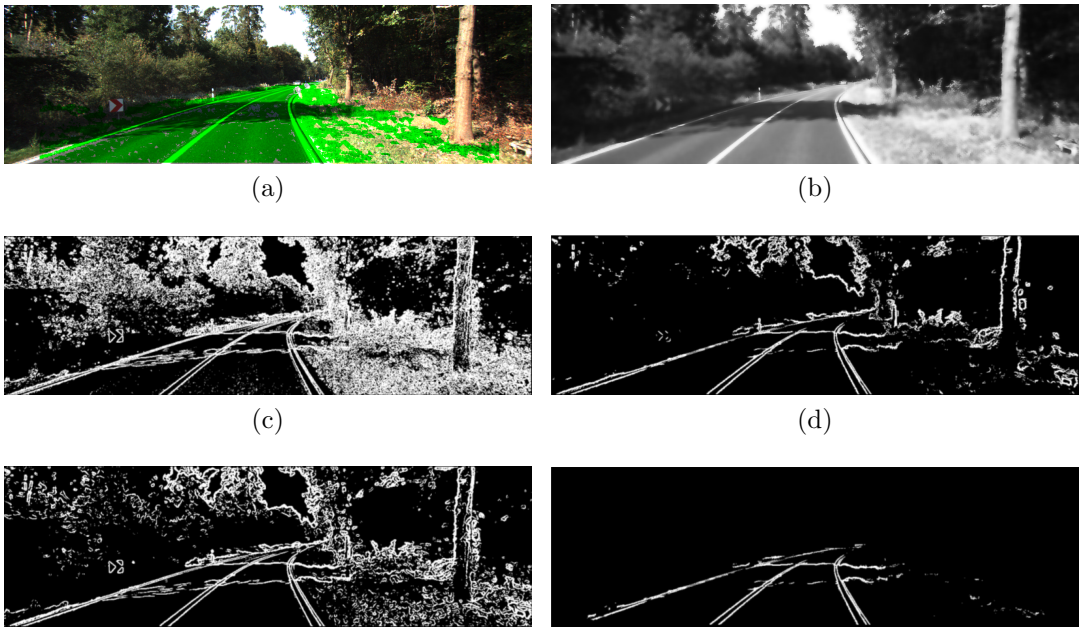
is updated, where tr_v is set to 4 in this chapter. Otherwise, it will be marked as an outlier and removed from $\boldsymbol{M_v}$. The iteration works until the percentage of the inliers exceeds a pre-set threshold ϵ_v, where ϵ_v is set to 99%. Finally, the candidates in the updated $\boldsymbol{M_v}$ are used to estimate the parameter vector $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2]^\top$. Compared with the parabola obtained in [6], the parabola estimation with the assistance of the RANSAC is more reliable and less affected by the outliers (an example is shown in Figure 4.2c). v_{vp} can be computed as follows:

$$v_{vp}(v) = v - \frac{\beta_0 + \beta_1 v + \beta_2 v^2}{\beta_1 + 2\beta_2 v} \tag{4.4}$$

4.2.3 Dense u_{vp} Estimation

4.2.3.1 Sparse u_{vp} Estimation

Before estimating u_{vp}, the road surface area is first estimated by comparing the difference between the actual and fitted disparity values. A pixel at (u, v) in the disparity map ℓ is considered to be in the road surface area if it satisfies the conditions $|\ell(u, v) - f(v)| \leq tr_{RSE}$ and $f^{-1}(0) \leq v \leq v_{max}$, where f^{-1} is the inverse function of $f(v) = \beta_0 + \beta_1 v + \beta_2 v^2$ and $tr_{RSE} = 3$ is a threshold set to remove the obstacles and potholes. The estimated road surface area is illustrated in green as shown in Figure 4.3a. This greatly reduces the unnecessary edge information used for dense u_{vp} estimation. The procedures in the later sections only focus on the road surface area.



(a) (b) (c) (d) (e) (f)

Figure 4.3: Sparse u_{vp} estimation. (a) road surface estimation. (b) bilateral filtering for Figure 3.4a. (c) edge detection result of Figure 3.4a. (d) edge detection result of (b). (e) edge detection result of the median filtering output. (f) edges in the road surface area. The green area in (a) illustrates the road surface. For the process of the bilateral filtering, σ_s and σ_r are empirically set to 300 and 0.3, respectively. The window sizes of the bilateral filter and the median filter are 11×11. The thresholds of the Sobel edge detection in (c), (d), (e) and (f) are 100. In the following procedures, only the edge pixels in (f) are considered.

Furthermore, the noise introduced in the imaging procedure makes the edge detectors such as Sobel very sensitive to the blobs [99]. Therefore, a bilateral filter is used to reduce the noise before detecting edges. Compared with the median filter which was utilised in [6], the bilateral filter is more capable of preserving edges when smoothing an image. The expression of a bilateral filter is shown as follows [100]:

$$i^{bf}(u,v) = \frac{\sum_{x=u-\varrho}^{x=u+\varrho} \sum_{y=v-\varrho}^{y=v+\varrho} \omega_s(x,y)\omega_r(x,y)i(x,y)}{\sum_{x=u-\varrho}^{x=u+\varrho} \sum_{y=v-\varrho}^{y=v+\varrho} \omega_s(x,y)\omega_r(x,y)} \tag{4.5}$$

where

$$\omega_s(x,y) = \exp\left\{-\frac{(x-u)^2+(y-v)^2}{\sigma_s^2}\right\}$$
$$\omega_r(x,y) = \exp\left\{-\frac{(i(x,y) - i(u,v))^2}{\sigma_r^2}\right\} \tag{4.6}$$

$i(x,y)$ is the intensity of the input image at (x,y) and $i^{bf}(u,v)$ is the intensity of the filtered image at (u,v). The block size of the filter is $(2\varrho+1)\times(2\varrho+1)$, and its centre is (u,v). The coefficients ω_s and ω_r are based on spatial distance and colour similarity, respectively. σ_s and σ_r are the parameters of ω_s and ω_r, respectively. In order to preserve only the edge information required for lane detection, σ_s and σ_r are set to 300 and 0.3, respectively. The output of bilateral filtering is shown in Figure 4.3b. The edge detection results of Figure 4.3b and Figure 3.4a are illustrated in Figure 4.3d and Figure 4.3c, respectively. As for the edge detection result of the median filtering output, it is depicted in Figure 4.3e. Obviously, although the median filter has removed a lot of redundant edges, the bilateral filter still achieves a better performance in terms of noise elimination and edge preservation.

In the following procedures, only the pixels in the road surface area are considered. The edge map in the road surface area is shown in Figure 4.3f. The sparse u_{vp} of each edge pixel $\boldsymbol{p_e} = [u_e, v_e]^\top$ can be estimated using the following equation:

$$u_{vp}^s(u_e, v_e) = u_e + \frac{v_e - v_{vp}(v_e)}{\nabla(\boldsymbol{p_e})} \tag{4.7}$$

where $\nabla(\boldsymbol{p_e}) = [g_u, g_v]^\top$ is the gradient of $\boldsymbol{p_e}$ that can be approximated using a Sobel operator, as shown in Eq.

$$g_u = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * i^{bf}, \quad g_v = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * i^{bf} \tag{4.8}$$

g_u and g_v represent the vertical and horizontal gradients of $\boldsymbol{p_e}$, respectively.

$u^s_{vp}(u_e, v_e)$ at the position of (u_e, v_e) is recorded in a 2-D sparse u_{vp} map. It is to be noted that u^s_{vp} represents sparse u_{vp} in this chapter.

Next, some details on the implementation are provided. As discussed in Section 2.7, the constant memory is read-only and beneficial for the data that will not change over the course of a kernel execution [92], therefore, two lookup tables are created on it to store the values of ω_s and ω_r, and the execution of the bilateral filtering can be highly accelerated. As for the implementation of the edge detection, due to the fact that the address of $i^{bf}(u, v)$ is always accessed repeatedly when determining whether a neighbour of $i^{bf}(u, v)$ belongs to an edge or not. Thus, a group of data i^{bf} is first loaded into the shared memory for each thread block. All threads within the same thread block will access the shared data instead of fetching them repeatedly from the global memory. In order to avoid the race conditions among different threads which run logically in parallel instead of executing physically concurrently, the threads within the same thread block need to be synchronised after they finish the data loading. Compared with the implementation on a Core-i7 4720HQ CPU processing with a single thread, the implementation on a GTX 970M GPU speeds up the execution of sparse u^s_{vp} estimation by over 74 times.

4.2.3.2 Dense u_{vp} Accumulation

To acquire u^d_{vp} information, the votes of u^s_{vp} within a rectangle of width $2w+1$ are accumulated, where w is set to 25 to accumulate as many votes as possible without compromising the execution speed. It is to be noted here that u^d_{vp} represents the dense u_{vp}. More details on the process of u^d_{px} accumulation are given in algorithm 5.

The initial step is to form a 1-D u^d_{vp} histogram by accumulating the votes from each edge pixel $\boldsymbol{p_e}$ on row $v_{max} - w$. In order to ensure energy minimisation rather than energy maximisation, the parameter m has to be a positive number and it is simply set to 1 in this chapter. Then, the votes of u^s_{vp} from each edge pixel $\boldsymbol{p_e}$ on row $v - 1$ are accumulated with the 1-D u^d_{vp} histogram on row v to form the 1-D u^d_{vp} histogram on row $v - 1$. This works until the width of the rectangle is able to reach $2w + 1$. Then, the current rectangle is shifted slightly

up to create another 1-D u^d_{vp} histogram. In order to improve the computational efficiency, SW algorithm is used to create 1-D u^d_{vp} histograms. The votes that appear on row $v - w$ above from the current rectangle are subtracted and those that appear on the bottom row of the previous rectangle are added. It is to be noted here that more u^s_{vp} votes correspond to a negatively higher value in the 2-D u^d_{vp} accumulator. This ensures the energy minimisation in the DP.

Furthermore, due to that the far field of the road may contain a higher lane curvature, a thinner rectangle is more desirable for the top rows of the image [6]. Therefore, only the votes on row $v + w + 1$ are added to update the 1-D u^d_{vp} histogram without the subtractions from the current rectangle. The SW algorithm makes the 1-D u^d_{vp} histogram update more efficiently by simply processing the bottom row and the top row. The result of dense u_{vp} accumulation is illustrated

Algorithm 5: Dense u_{vp} accumulation.

Input : 2-D u^s_{vp} map
Output: 2-D u^d_{vp} accumulator

set all elements in u^d_{vp} accumulator to 0;
for $v \leftarrow v_{max} - 2w$ *to* v_{max} **do**
  for $u \leftarrow u_{min}$ *to* u_{max} **do**
    $u^d_{vp}(u^s_{vp}(u, v), v_{max} - w) \leftarrow u^d_{vp}(u^s_{vp}(u, v), v_{max} - w) - m$
  end
end
for $v \leftarrow v_{max} - w - 1$ *to* $f^{-1}(0) + w$ **do**
  for $u \leftarrow u_{min}$ *to* u_{max} **do**
    $u^d_{vp}(u, v) \leftarrow u^d_{vp}(u, v + 1)$;
    $u^d_{vp}(u^s_{vp}(u, v - w), v) \leftarrow u^d_{vp}(u^s_{vp}(u, v - w), v) - m$;
    $u^d_{vp}(u^s_{vp}(u, v + w + 1), v) \leftarrow u^d_{vp}(u^s_{vp}(u, v + w + 1), v) + m$;
  end
end
for $v \leftarrow f^{-1}(0) + w - 1$ *to* $f^{-1}(0)$ **do**
  for $u \leftarrow u_{min}$ *to* u_{max} **do**
    $u^d_{vp}(u, v) \leftarrow u^d_{vp}(u, v + 1)$;
    $u^d_{vp}(u^s_{vp}(u, v + w + 1), v) \leftarrow u^d_{vp}(u^s_{vp}(u, v + w + 1), v) + m$;
  end
end

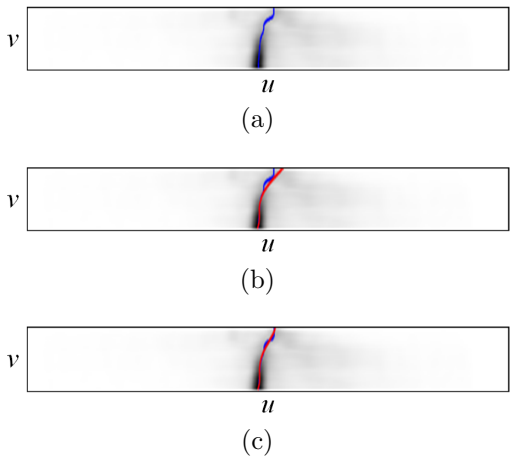


Figure 4.4: Dense u_{vp} accumulation and estimation. (a) dense u_{vp} accumulator. (b) target solution obtained in [6]. (c) target solution obtained in the proposed system. The blue paths are the optimal solutions obtained using the DP. $g(v) = \gamma_0 + \gamma_1 v + \gamma_2 v^2 + \gamma_3 v^3 + \gamma_4 v^4$ is plotted in red.

in Figure 4.4, where $m_u(u, v)$ represents the votes of $u_{vp} = u$ on row v.

4.2.3.3 u_{vp} estimation

Similarly, u^d_{vp} accumulator is optimised using the DP to extract the path with the minimal energy, as shown in Eq. 4.1. In the first iteration, $E_{smooth} = 0$ and $E_{data} = m_u(u, v_{max})$. Then, E is computed based on the previous iterations:

$$E(u)_v = m_u(u, v) + \min_{\tau_u}[E(u_{vp} + \tau_u)_{v+1} + \lambda_u \tau_u], \text{ s.t. } \tau_u \in [-5, 5] \tag{4.9}$$

The solution $\boldsymbol{M_u} = [\boldsymbol{u}, \boldsymbol{v}]^\top \in \mathbb{R}^{t\times 2}$ with the minimal energy is then selected as the optima, which is plotted in blue, as shown in Figure 4.4. The blue path includes t points. The two column vectors $\boldsymbol{u} = [u_0, u_1, \ldots, u_{t-1}]^\top$ and $\boldsymbol{v} = [v_0, v_1, \ldots, v_{t-1}]^\top$ record the column and row numbers, respectively. The parameter vector $\boldsymbol{\gamma} = [\gamma_0, \gamma_1, \gamma_2, \gamma_3, \gamma_4]^\top$ can be estimated by solving the least squares problem in Eq. 4.10. Here, the author uses the same strategy as the estimation of $\boldsymbol{\beta}$. $\mathscr{I}$ and $\boldsymbol{M_u}$ are updated using the RANSAC until the percentage of the inliers exceeds a pre-set threshold. Then, $\boldsymbol{\gamma}$ is obtained by fitting a quartic

polynomial to the candidates in the updated $\boldsymbol{M_u}$. Algorithm 6 provides more details on $\boldsymbol{\gamma}$ estimation.

$$\boldsymbol{\gamma} = \arg\min_{\gamma} \sum_{j=0}^{t-1} (u_j - (\gamma_0 + \gamma_1 v_j + \gamma_2 {v_j}^2 + \gamma_3 {v_j}^3 + \gamma_4 {v_j}^4))^2 \tag{4.10}$$

Algorithm 6: $\boldsymbol{\gamma}$ estimation with the assistance of the RANSAC.

Input : optimal solution: $\boldsymbol{M_u} = [\boldsymbol{u}, \boldsymbol{v}]^\top$
Output: parameter vector: $\boldsymbol{\gamma}$

do
- randomly select a specified number of candidates $[u_j, v_j]^\top$;
- fit a quartic polynomial to the selected candidates and get $\boldsymbol{\gamma}$;
- determine the numbers of inliers and outliers: $n_{\mathscr{I}}$ and $n_{\mathscr{O}}$, respectively;
- remove the outliers from $\boldsymbol{M_u}$;

while $n_{\mathscr{I}}/(n_{\mathscr{I}} + n_{\mathscr{O}}) < \epsilon_u$;
fit a quartic polynomial to the candidates in the updated $\boldsymbol{M_u}$ and get $\boldsymbol{\gamma}$;

To determine whether a given candidate $[u_j, v_j]^\top$ belongs to $\mathscr{I}$, the corresponding squared residual $r_j = (u_j - g(v_j))^2$ needs to be computed. If r_j is smaller than a pre-set threshold tr_u, the candidate is marked as an inlier and $\mathscr{I}$ is updated, where tr_u is set to 16 in this chapter. Otherwise, it will be marked as an outlier and removed from $\boldsymbol{M_u}$. The iteration works until the percentage of the inliers exceeds the pre-set threshold ϵ_u, where ϵ_u is set to 99% in this chapter. Finally, $\boldsymbol{\gamma}$ can be estimated by fitting a quartic polynomial to the updated $\boldsymbol{M_u}$. Compared with the target solution obtained in [6], as shown in Figure 4.4b, the target solution obtained in the proposed system is less affected by the outliers (an example is shown in Figure 4.4c). In practical implementation, Eq. 4.10 is rearranged as shown in Eq. 4.11 to avoid the data overflow when fitting the quartic polynomial.

$$\boldsymbol{\gamma} = (\kappa_0 \boldsymbol{P}^\top \boldsymbol{P})^{-1} (\kappa_0 \boldsymbol{P}^\top) \boldsymbol{u} \tag{4.11}$$

where

$$\boldsymbol{P} = \begin{bmatrix} 1 & v_0 & \cdots & {v_0}^4 \\ 1 & v_1 & \cdots & {v_1}^4 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & v_{t-1} & \cdots & {v_{t-1}}^4 \end{bmatrix} \tag{4.12}$$

$\boldsymbol{P}$ is a Vandermonde matrix. κ_0 is used to avoid the data overflow problem caused by the higher order polynomials (e.g., when $v = 375$, $v^8 \approx 4 \times 10^{20}$ which is far beyond the significand range of *long double* type in C language). After estimating $\boldsymbol{\gamma}$, u_{vp} can be computed as follows:

$$u_{vp}(v) = \gamma_0 + \gamma_1 v + \gamma_2 v^2 + \gamma_3 v^3 + \gamma_4 v^4 \tag{4.13}$$

4.2.4 Lane Position Validation

$\boldsymbol{p_{vp}}$ provides the tangential direction and the curvature information of lanes, which can help to validate the lane positions. In [6], the authors formed a likelihood function $V(\boldsymbol{p_e}) = \nabla(\boldsymbol{p_e}) \cdot \cos(\theta_{\boldsymbol{p_e}} - \theta_{\boldsymbol{p_{vp}}})$ for each edge point $\boldsymbol{p_e}$ and selected the plus-minus peak pairs for visualisation, where $\theta_{\boldsymbol{p_e}}$ is the angle between the u-axis and the orientation of the edge point $\boldsymbol{p_e}$, and $\theta_{\boldsymbol{p_{vp}}}$ is the angle between the u-axis and the radial from an edge pixel $\boldsymbol{p_e}$ to $\boldsymbol{p_{vp}}(v_e)$. Although the peak pair selection algorithm presented in [6] has achieved some impressive experimental results, some inaccurate detections still occurred. In this chapter, the energy of each possible solution is computed and all satisfying lane positions are selected for visualisation. Algorithm 7 provides more details on the proposed approach.

For the dark-light transition of a lane marking, the value of g_u is positive and higher than the ones at the non-edge positions. As for the light-dark transition of a lane marking, the value of g_u is negative but its absolute value is still higher than the ones at the non-edge positions [6]. Therefore, the task to validate lane positions now only involves the estimation of the centre position of each pair of dark-light and light-dark transitions. To reduce g_u accumulation from the

Algorithm 7: Lane position validation.

Input : $\boldsymbol{p_{vp}}$ and g_u
Output: lane position vector: $\boldsymbol{\delta}$

create two 2-D maps $\mathcal{M}_0$, $\mathcal{M}_1$ with the same size as the input image i;
set all elements in $\mathcal{M}_0$, $\mathcal{M}_1$ to 0;
create a 1-D histogram $\mathcal{H}$ of size $(2t+1)u_{max}$;
$\forall \mathcal{M}_0(u,v) \leftarrow \sum_{x=-\kappa_1}^{x=+\kappa_1} \sum_{y=-\kappa_2}^{y=+\kappa_2} g_u(u+x, v+y)\omega_g(u+x, v+y)$;
approximate the horizontal gradient of each point in $\mathcal{M}_0$ using the Sobel operator and save the results in $\mathcal{M}_1$;
for $k \leftarrow -tu_{max}$ *to* tu_{max} **do**
    aggregate $\mathcal{M}_1$ from row v_{max} to row $f^{-1}(0)$ to get the energy E;
    $\mathcal{H}(k + tu_{max}) \leftarrow E$;
end
if $\mathcal{H}(k) < \min\{\mathcal{H}(k-1), \mathcal{H}(k+1)\}$ **then**
    put $k - tu_{max}$ into $\boldsymbol{\delta}$;
end
remove the elements which are smaller than the threshold tr_{LPV} from $\boldsymbol{\delta}$;
remove the nearby candidates from $\boldsymbol{\delta}$;
multiple lanes visualisation;

non-lane edges, a piecewise weighting ω_g is proposed as follows:

$$\omega_g(u_e, v_e) = \begin{cases} \exp\left(-\frac{|\theta_{\boldsymbol{p_e}} - \theta_{\boldsymbol{V_p}}|}{{\sigma_g}^2} \cdot \frac{1}{\theta_s}\right), & |\theta_{\boldsymbol{p_e}} - \theta_{\boldsymbol{V_p}}| \leq \frac{\pi}{6} \\ 0, & \text{otherwise} \end{cases} \tag{4.14}$$

where the step θ_s is set to $\pi/36$. The portion $|\theta_{\boldsymbol{p_e}} - \theta_{\boldsymbol{p_{vp}}}|/\theta_s$ is used to provide a Gaussian weight so as to decrease the magnitude of noise pixels, where σ_g is set to 3.5 to ensure that the value of ω_g can reach around 0.85 when $|\theta_{\boldsymbol{p_e}} - \theta_{\boldsymbol{p_{vp}}}| = \pi/2$. Then, the values of $g_u\omega_g$ are summed within a shifting box to further reduce the noise, where the box size is $(2\kappa_1+1) \times (2\kappa_2+1)$. Since a larger box size requires more processing time, κ_1 and κ_2 are empirically set to 1 and 3, respectively. The accumulation output is then saved in a 2-D map $\mathcal{M}_0$, where the horizontal gradient from a dark-light peak to a light-dark peak is negative. To approximate the horizontal gradients, the Sobel horizontal kernel is convoluted with $\mathcal{M}_0$ and the convolution result is saved in $\mathcal{M}_1$. Then, the values of $\mathcal{M}_1$ are aggregated for each possible solution from row v_{max} to row $f^{-1}(0)$ as follows:

$$E(v)_{u_v} = \mathcal{M}_1(u_v, v) + \lambda_g E(v+1)_{u_{v+1}} \tag{4.15}$$

where

$$u_v = \frac{u_{vp}(v+1) + v u_{v+1} - v_{vp}(v+1)u_{v+1}}{v + 1 - v_{vp}(v+1)} \tag{4.16}$$

In order to find all possible lane positions, t is set to 0.5. This implies that u starts from $-0.5u_{max}$ and ends at $1.5u_{max}$. In the first iteration, a possible position $(u_{v_{max}}, v_{max})$ is selected and the total energy E is simply set to $\mathcal{M}_1(u_{v_{max}}, v_{max})$. Then, $\boldsymbol{p_{vp}}$ information is used to estimate the next position $(u_{v_{max}-1}, v_{max} - 1)$ on the selected track. The energy E is updated using Eq. 4.15. Here, λ_g has been used for test purpose and the value of 1 has been found to provide the good results during the experiments. The aggregation of $\mathcal{M}_1$ works until v reaches $f^{-1}(0)$. For each lane starting from $(u_{v_{max}}, v_{max})$, its total energy is saved in a 1-D histogram $\mathcal{H}$. Then, the local minima which are smaller than a pre-set threshold tr_{LPV} are picked out. At the same time, if two local minima are quite close to each other, the minima with a larger energy is ignored. Finally, the lanes can be visualised by iterating Eq. 4.16. The lane detection results will be discussed in section 4.3.

4.3 Experimental Results

Currently, it is impossible to access a satisfying ground truth dataset for the evaluation of lane detection algorithms because accepted test protocols do not usually exist [101]. Therefore, many publications related to lane detection only focus on the quality of their experimental results [6]. For this reason, the author compares the performance of the proposed system with [6] and [74] in terms of both accuracy and speed. Some successful detection examples are shown in Figure 4.5.

Firstly, the accuracy of the proposed algorithm is evaluated. The lane detection results of the proposed algorithm and the algorithms described in [6] and [74] are detailed in Table 4.1, Table 4.2 and Table 4.3, respectively. To evaluate the robustness of the proposed algorithm, eight sequences are selected from the KITTI database (including two additional sequences) for tests: 2495 frames

containing 5361 lanes [35] (1637 lanes more than what were used in [6] and [74]). The image resolution is 1242×375 in sequences 1 to 6, 1241×376 in sequence 7, and 1238×374 in sequence 8. From Table 4.1, it can be seen that the proposed algorithm presents a better detection ratio, where 99.9% lanes are successfully detected in sequences 1 to 7 (including all the sequences in Table 4.2 and Table 4.3), while the detection ratios of [6] and [74] are only 98.7% and 89.2%, respectively. The comparison between some failed examples in [6] and the corresponding results obtained using the proposed algorithm is illustrated in Figure 4.6.

Figure 4.5: Lane detection results. The red lines illustrate the detected lanes.

Table 4.1: Detection results of the proposed algorithm.

Sequence	Lanes	Incorrect detection	Misdetection
1	860	0	0
2	594	0	0
3	376	0	0
4	156	0	0
5	678	0	0
6	1060	1	2
7	644	0	0
8	993	18	7
Total	5361	19	9

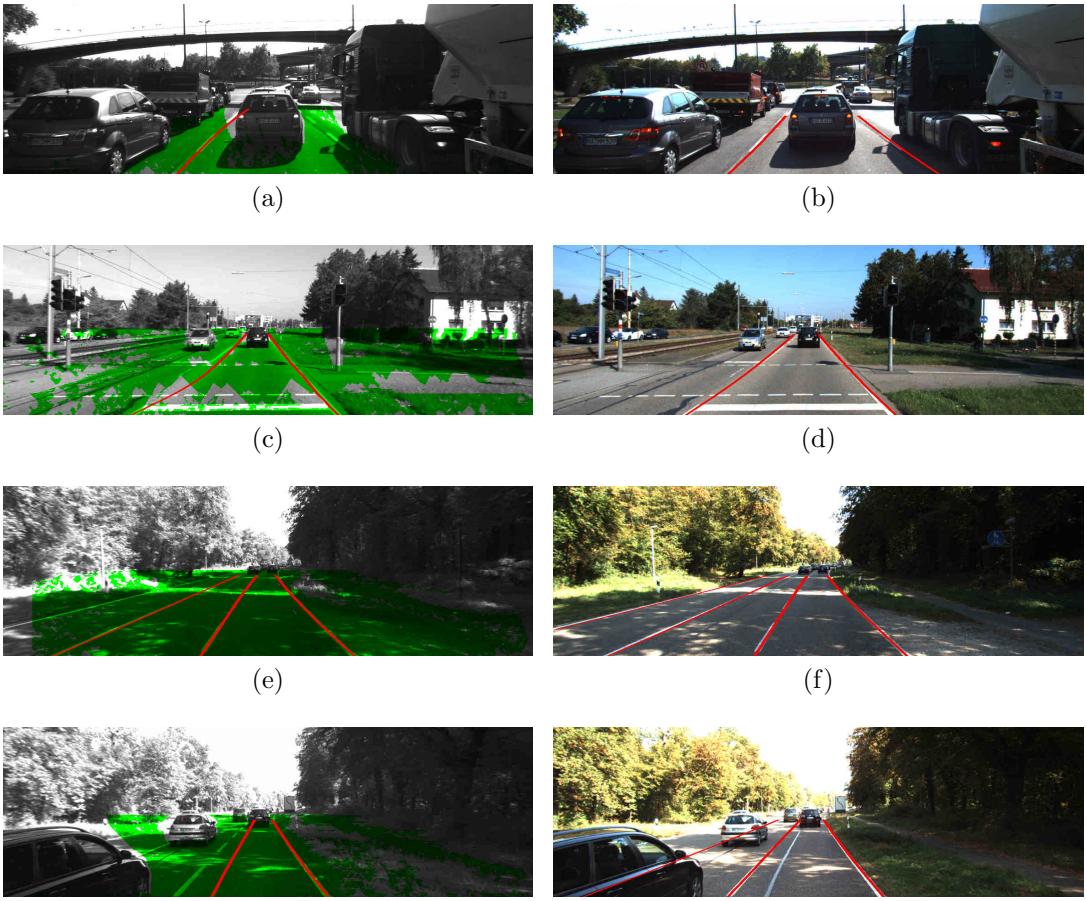
Table 4.2: Detection results of [6].

Sequence	Lanes	Incorrect detection	Misdetection
1	860	0	0
2	594	0	0
3	376	0	0
4	156	0	9
5	678	0	17
6	1060	14	7
Total	3724	14	33

Table 4.3: Detection results of [74].

Sequence	Lanes	Incorrect detection	Misdetection
1	860	12	0
2	594	44	0
3	376	44	0
4	156	17	0
5	678	107	0
6	1060	180	0
Total	3724	404	0

In Figure 4.6a, it can be seen that the obstacle areas occupy a larger portion than the road surface area, which severely affects the accuracy of v_{vp} estimation. When both inliers and outliers are used in the LSF, $\boldsymbol{p_{vp}}$ differs too much from the ground truth. This further influences the lane position validation and leads to an



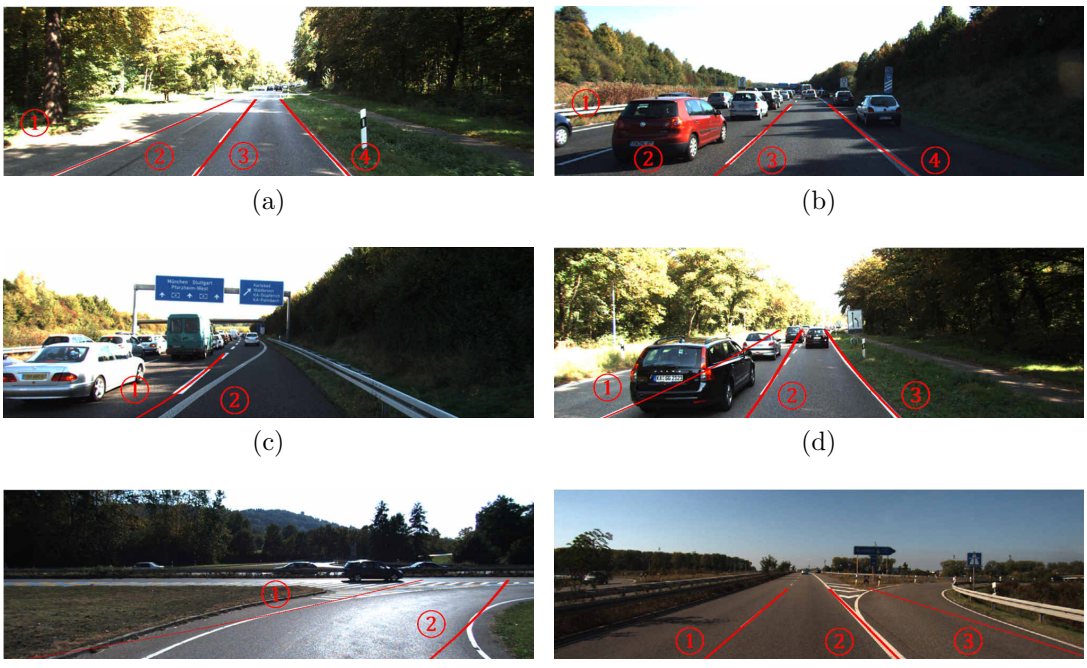
(a) (b) (c) (d) (e) (f) (g) (h)

Figure 4.6: Comparison between some failed examples in [6] and the corresponding results in this chapter. The green areas in the first column illustrate the road surface. The red lines are the detected lanes. The first column illustrates the failed examples in [6], and the second column shows the corresponding results of the proposed system.

imprecise detection and a misdetection. In Figure 4.6b, it can be seen that the LSF considering only inliers increases the precision of v_{vp} estimation significantly. Moving to the second row, an over-curved lane can be seen in Figure 4.6c. When we estimate $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ with the assistance of the RANSAC, the improvement can be observed in Figure 4.6d, where a more reasonable lane is detected. In Figure 4.6e, the lane near the left road boundary is misdetected because the low contrast between lane and road surface reduces its magnitude in the stage of lane position validation. In Figure 4.6g, an incorrect detection occurs because a road marking

is more contrastive to the road surface. In Section 4.2.4, a more effective piecewise weighting ω_g is proposed to update g_u for the edge pixels. Then, g_u of the non-lane edges reduces significantly, which therefore greatly helps the system to avoid the incorrect detections of some lane markings. Also, $g_u\omega_g$ within a shifting box is summed for each position. This increases the magnitude of the lanes which are lowly contrastive to the road surface. The misdetections in Figure 4.6e and Figure 4.6g are thus detectable, and the failed detection in Figure 4.6g is also corrected. The corresponding results are shown in Figure 4.6f and Figure 4.6h.

In practical experiments, the failed cases consist of misdetections and incorrect detections. The misdetections are mainly caused by: image over-exposure, partially occluded by the obstacles, forks on the road. The corresponding examples are illustrated in Figure 4.7a, Figure 4.7b and Figure 4.7c, respectively. In Figure 4.7a, due to the image over-exposure, the edges pixels on lane 1 are rare, which leads to its misdetection. In Figure 4.7b, it can be seen that the vehicles partially occlude lane 1 and lane 2. The occlusion decreases the absolute value



(a) (b) (c) (d) (e) (f)

Figure 4.7: Examples of the failed detections in this chapter.

of $g_u\omega_g$ when trying to validate the lane positions, which makes lane 1 and lane 2 undetectable. In Figure 4.7c, lane 2 forks from lane 1, and thus, has a different curvature information from lane 1, which therefore causes the misdetection.

For the factors leading to incorrect detections, the author groups them into three main categories: ambiguous disparity projection of a road surface on the v-disparity map; $\boldsymbol{p_{vp}}$ that does not exist in the image; different roadways, which are presented in Figure 4.7d, Figure 4.7e and Figure 4.7f, respectively. In Figure 4.7d, the obstacles, e.g., vehicles and trees, take a big portion in the image. Therefore, when d is around 0, m_v is mainly accumulated by the pixels on the obstacles and sky, which affects the accuracy of v_{vp} estimation. This further makes the detected lane markings slightly above the ground truth when they move to the boundary between the road surface and sky. In Figure 4.7e, $\boldsymbol{p_{vp}}$ does not exist, which affects the detection results. In Figure 4.7f, there are two different roadways: roadway between lane 1 and lane 2; roadway between lane 2 and lane 3. The second roadway turns right and therefore has a different $\boldsymbol{p_{vp}}$ from the first roadway, which leads to an imprecise detection of lane 3.

Finally, the processing speed is discussed. The algorithm is implemented on a heterogeneous system consisting of an Intel Core i7-4720HQ CPU and an NVIDIA GTX 970M GPU. The GPU has 10 Streaming Multiprocessors with 128 CUDA cores on each of them. The runtime of the proposed system is around 7 ms (excluding the runtime of the disparity estimation), which is approximately 38 times faster than the lane detection algorithm proposed in [6] where 263 ms was achieved on an Intel i5-5300U CPU (frequency: 2.3GHz, the number of cores: 2). The authors believe that the failed cases can be reduced in the future by adding a lane tracking algorithm. The demo videos are available at: `http://www.ruirangerfan.com`

4.4 Conclusion

A multiple lane detection system was presented in this chapter. The novelties include: an improved dense vanishing point estimation method, a novel lane position validation algorithm and a real-time implementation on a heterogeneous system. To evaluate the performance, 5361 lanes from eight datasets were tested.

The experimental results illustrate that the proposed algorithm works more accurately and robustly than the state-of-the-art lane detection algorithm presented in [6]. By highly exploiting the GPU architecture and allocating different parts of the proposed algorithm on different platforms for execution, a high processing speed of 143 fps was achieved.

Chapter 5

Road Surface 3-D Reconstruction Based on Dense Subpixel Disparity Map Estimation

Various 3-D reconstruction methods have enabled civil engineers to detect damage on a road surface. To achieve the millimetre accuracy required for road condition assessment, a disparity map with subpixel resolution needs to be used. However, none of the existing stereo matching algorithms are specially suitable for the reconstruction of the road surface. Hence in this chapter, a novel dense subpixel disparity estimation algorithm with high computational efficiency and robustness is proposed. This is achieved by first transforming the perspective view of the target frame into the reference view, which not only increases the accuracy of the block matching for the road surface but also improves the processing speed. The disparities are then estimated using the algorithm presented in Chapter 3. Since the search range is obtained from the previous iteration, errors may occur when the propagated search range is not sufficient. Therefore, a correlation maxima verification is performed to rectify this issue, and the subpixel resolution is achieved by conducting a parabola interpolation enhancement. Furthermore, a novel disparity global refinement approach developed from the Markov Random Fields and Fast Bilateral Stereo is introduced to further improve the accuracy of the estimated disparity map, where disparities are updated

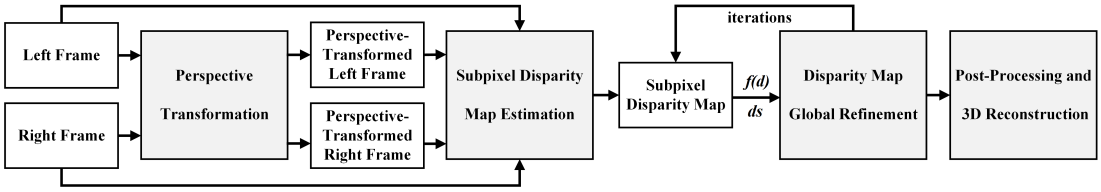


Figure 5.1: Stereo vision-based road surface 3-D reconstruction system workflow.

iteratively by minimising the energy function that is related to their interpolated correlation polynomials. The algorithm is implemented in C language with a near real-time performance. The experimental results illustrate that the absolute error of the reconstruction varies from 0.1 mm to 3 mm. The main contributions of this chapter are published in [40].

5.1 System Overview

The aim of this chapter is to reconstruct the road scenes for pothole detection. In this regard, the proposed disparity estimation algorithm is developed from the work presented in Chapter 3. To assess the condition of a road surface, millimetre accuracy is desired in 3-D reconstruction and thus disparities in subpixel resolution are inevitable. Therefore, the correlation costs around the initial disparity are interpolated into a parabola and the position of the extrema is selected as the subpixel disparity. However, the subpixel disparity maps obtained from parabola interpolation are still unsatisfactory because the correlation costs of neighbourhood systems are not aggregated before finding the best disparities. To aggregate neighbouring costs adaptively, some authors have proposed to filter the whole cost volume with a bilateral filter since it provides a feasible solution for the initial message passing problem on a fully connected MRF [18]. These algorithms are also known as FBS [54–56]. However, the intensive computational complexity introduced when filtering the whole cost volume severely impacts on the processing speed. In this regard, the author believes that only the candidates around the best disparities need to be processed and a novel disparity refinement approach is proposed in this work. The workflow of the proposed road surface 3-D reconstruction system is depicted in Figure 5.1. Firstly, the perspective view of the road

surface in the target image is transformed into its reference view, which greatly enhances the similarity of the road surface between the two images. Since the propagated search range is sometimes insufficient, the desirable disparities have to be further verified to ensure they possess the highest correlation costs. The latter ensures the feasibility of parabola interpolation-based subpixel enhancement. To further optimise the obtained subpixel disparity map, the interpolated parabola functions $f(d)$ are set as the labels in the MRF because they contain the information of both disparity values and correlation costs. By updating the parabola functions $f(d)$ and subpixel disparities d_s iteratively, a disparity in a continuous area becomes smooth but it is preserved when discontinuities occur. Finally, each 3-D point on the road surface is computed based on its projections on the left and right images. The reconstruction accuracy is evaluated using three sample models (see subsection 5.3.1 for more details). The datasets are publicly available at: http://www.ruirangerfan.com.

The rest of the chapter is structured as follows: Section 5.2.1 presents a novel perspective transformation (PT) method. Section 5.2.2 describes a subpixel disparity estimation algorithm. A disparity map global refinement approach is introduced in section 5.2.3. In Section 5.2.4, the disparity map is post-processed and the 3-D road surface is reconstructed. In Section 5.3, the experimental results are illustrated and the performance of the proposed algorithm is evaluated. Finally, Section 5.4 summarises the chapter.

5.2 Algorithm Description

5.2.1 Perspective Transformation

In this chapter, the proposed algorithm focuses entirely on the road surface which can be treated as a GP. To enhance the accuracy of stereo matching, the author first draws on the concept of ground plane constraint in [102] and [103] to transform the perspective views of two images before estimating their disparities. GP constraint is commonly used in a wide range of obstacle detection systems, where the image on one side is set as the reference and the other image is transformed into the reference view. Pixels arising from the GP satisfy the same affine trans-

formation while an object above the GP will not be transformed successfully [102]. Referring to the experimental results in [103], pixels from an obstacle are distorted in the transformed image. Nevertheless, the GP in the transformed image looks more similar to its reference view. Therefore, a perspective transformation makes the obstacle areas noisy and unreliable but greatly enhances the similarity of the road surface between two images.

As discussed in Section 2.2.5.3, a homograph matrix $\boldsymbol{H}$ can be used to distinguish obstacles from the ground plane [102]. Generally, $\boldsymbol{H}$ can be estimated with at least four pairs of correspondences $\boldsymbol{p_l} = [u_l, v_l]^\top$ and $\boldsymbol{p_r} = [u_r, v_r]^\top$ [8]. Hattori et al. proposed a pseudo-projective camera model where several assumptions are made about road geometry to simplify the estimation of $\boldsymbol{H}$ [102]. In this chapter, the author improves on their algorithm by considering the following hypotheses:

- $\boldsymbol{K_l}$ and $\boldsymbol{K_r}$ in Eq. 2.25 are identical.
- $\boldsymbol{R}$ in Eq. 2.25 is an identity matrix.
- $\boldsymbol{t}$ in Eq. 2.25 is in the same direction as the $X^{\mathscr{W}}$-axis.
- the road surface in Eq. 2.23 is a horizontal plane: $n_1 Y^{\mathscr{W}} + \beta = 0$.
- rotation of the stereo rig is only about the $X^{\mathscr{W}}$-axis.

For a perfectly-calibrated stereo rig, $v_l = v_r = v$. The disparity is defined as $d = u_l - u_r$. The projection of a horizontal plane on the v-disparity map is a linear pattern [50]:

$$d = -\frac{T_c n_1}{\beta}(f \sin\theta - v_0 \cos\theta) - v\frac{T_c n_1}{\beta}\cos\theta = \alpha_0 + \alpha_1 v \tag{5.1}$$

where θ is the pitch angle between the stereo rig and the road surface (an example can be seen in Figure 5.6a), f is the focus length of the cameras, T_c is the baseline, and (u_0, v_0) is the principal point in pixel. When $\theta = \pi/2$, $d = -fT_c n_1/\beta$ is a constant. Otherwise, d is proportional to v [50]. This implies that a perspective distortion always exists for the GP in two images, which further affects the accuracy of block matching. Therefore, the PT aims to make the GP in the transformed image similar to that in the reference frame.

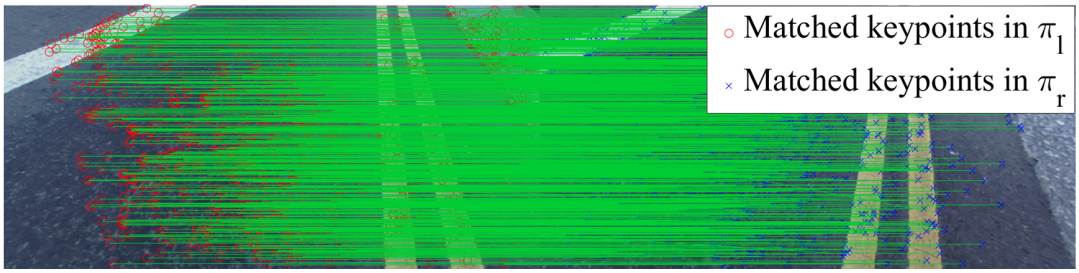


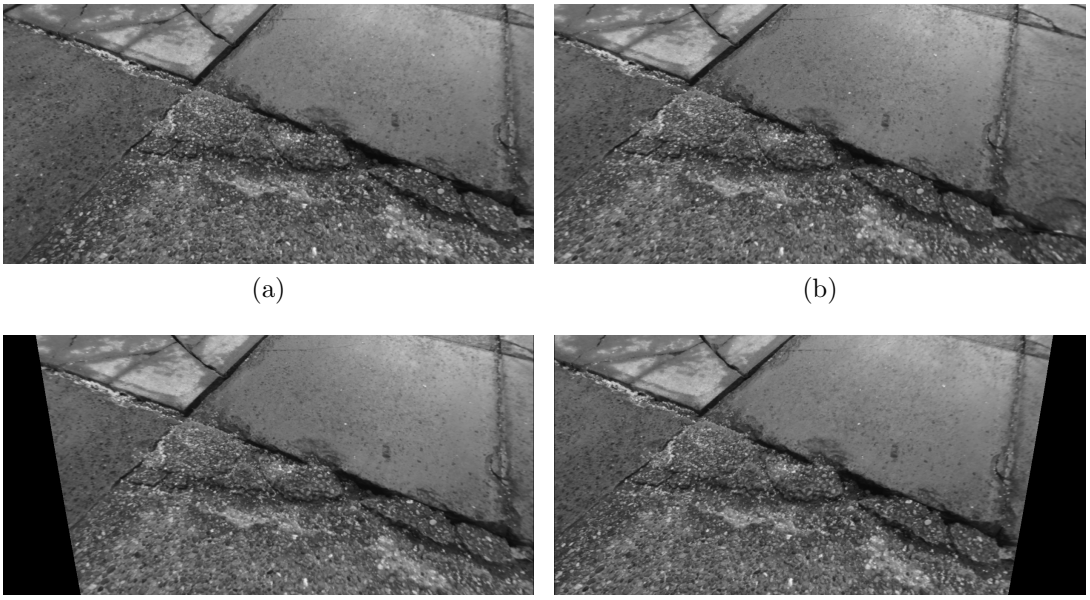
Figure 5.2: BRISK-based on-road keypoints detection and matching between the left and right images.

Algorithm 8: Perspective transformation.

Data: π_l and π_r
Result: $\boldsymbol{\alpha} = [\alpha_0, \alpha_1]^\top$
detect and match the keypoints in the left and right images;
if $|v_{li} - v_{ri}| > \epsilon$ *or* $u_{li} - u_{ri} < 0$ **then**
  remove $\boldsymbol{p_{li}}$ and $\boldsymbol{p_{ri}}$ from $\boldsymbol{Q_l}$ and $\boldsymbol{Q_r}$, respectively;
estimate $\boldsymbol{\alpha}$ using the least squares fitting;
all points in the target image are shifted $\alpha_0 + \alpha_1 v - \delta$ pixels to the reference view;

Now, the PT can be straightforwardly realised using parameters $\boldsymbol{\alpha} = [\alpha_0, \alpha_1]^\top$. The proposed PT is detailed in algorithm 8. $\boldsymbol{\alpha}$ can be estimated by solving a least squares problem with a set of reliable correspondences $\boldsymbol{Q_l} = [\boldsymbol{p_{l1}}, \boldsymbol{p_{l2}}, \ldots, \boldsymbol{p_{lm}}]^\top$ and $\boldsymbol{Q_r} = [\boldsymbol{p_{r1}}, \boldsymbol{p_{r2}}, \ldots, \boldsymbol{p_{rm}}]^\top$. In this chapter, Binary Robust Invariant Scalable Keypoints (BRISK) is utilised to detect and match $\boldsymbol{Q_l}$ and $\boldsymbol{Q_r}$. It allows a faster execution to achieve approximately the same number of correspondences as Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) [104]. An example of on-road keypoints detection and matching is illustrated in Figure 5.2.

Since outliers can severely affect the accuracy of least squares fitting, the less reliable correspondences are first removed before estimating $\boldsymbol{\alpha}$, where ϵ is proposed to be 1. For the left disparity map ℓ^{lf} estimation, each point on row v in π_r is shifted $\alpha_0 + \alpha_1 v - \delta$ pixels to the right, where δ is a constant set to 20 (for dataset 1 and 2) or 30 (for dataset 3) to guarantee that all the disparities are positive. Similarly, each point in π_l is shifted $\alpha_0 + \alpha_1 v - \delta$ pixels to the left



(a) (b)

(c) (d)

Figure 5.3: Perspective transformation. (a) left image. (b) right image. (c) transformed right image. (d) transformed left image. (a) and (c) are used as the input left and right images for the left disparity map estimation. (d) and (b) are used as the input left and right images for the right disparity map estimation.

when π_r is served as the reference. An example of perspective transformation is presented in Figure 5.3. The performance improvements achieved by using the PT will be discussed in Section 5.3.

5.2.2 Subpixel Disparity Map Estimation

As compared to many other stereo matching algorithms which aim at automotive applications, the trade-off between speed and precision has been greatly improved in the stereo vision system presented in Chapter 3. The left and right disparity maps, ℓ^{lf} and ℓ^{rt} are shown in Figure 5.4a and Figure 5.4b. The left disparity map after the LRC check processing is illustrated in Figure 5.4c.

5.2.2.1 CMV

Since the search range propagates using Eq. 3.7, errors may occur in subpixel enhancement when $c_{NCC}(u, v, d-1)$ or $c_{NCC}(u, v, d+1)$ is not computed and compared with $c_{NCC}(u, v, d)$. Therefore, CMV will run until the correlation cost of the disparity is a local maxima. More details are provided in algorithm 9.

Algorithm 9: Correlation maxima verification

Data: disparity map ℓ
Result: correlation maxima verified disparity map ℓ_{cmv}

if $c_{NCC}(u, v, d) > \max\{c_{NCC}(u, v, d-1), c_{NCC}(u, v, d+1)\}$ **then**
  $\ell_{cmv}(u, v) \leftarrow \ell(u, v)$;
else if $c_{NCC}(u, v, d-1) < c_{NCC}(u, v, d) < c_{NCC}(u, v, d+1)$ **then**
  repeat
    compute $c_{NCC}(u, v, d+k),\ k \geq 2$;
  until $c_{NCC}(u, v, d+k) < c_{NCC}(u, v, d+k-1)$;
  $\ell_{cmv}(u, v) \leftarrow d+k-1$;
else
  repeat
    compute $c_{NCC}(u, v, d-k),\ k \geq 2$;
  until $c_{NCC}(u, v, d-k) < c_{NCC}(u, v, d-k+1)$;
  $\ell_{cmv}(u, v) \leftarrow d-k+1$;
end

5.2.2.2 Subpixel Enhancement

In this chapter, the road surface application requires a millimetre accuracy in 3-D reconstruction. A disparity error larger than one pixel may result in a non-neglected difference in the reconstructed road surface [105]. Therefore, subpixel resolution is inevitable to achieve a highly accurate result.

For each pixel whose disparity d is $\ell(u, v)$, a parabola is fitted to three correlation costs $c_{NCC}(u, v, d-1)$, $c_{NCC}(u, v, d)$ and $c_{NCC}(u, v, d+1)$ around the initial disparity d. The centreline of the parabola is selected as the subpixel displacement d_s as follows [56]:

$$d_s = d + \frac{c_{NCC}(u, v, d-1) - c_{NCC}(u, v, d+1)}{2c_{NCC}(u, v, d-1) + 2c_{NCC}(u, v, d+1) - 4c_{NCC}(u, v, d)} \tag{5.2}$$

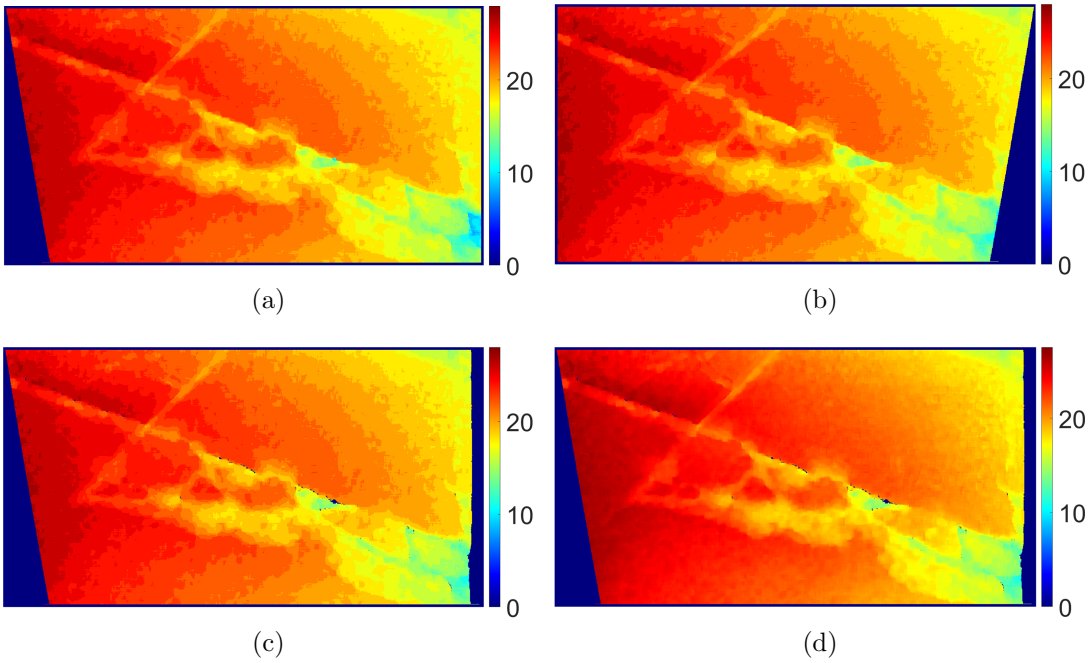


Figure 5.4: Subpixel disparity map estimation. (a) left disparity map. (b) right disparity map. (c) left disparity map processed with the LRC check. (d) subpixel disparity map.

Since the CMV guarantees that $c_{NCC}(u, v, d)$ is larger than both $c_{NCC}(u, v, d-1)$ and $c_{NCC}(u, v, d+1)$, d_s will be between $d-1$ and $d+1$. Figure 5.4c after the subpixel enhancement is given in Figure 5.4.

5.2.3 Disparity Map Global Refinement

In this chapter, the local algorithm proposed in Section 5.2.2 greatly minimises the trade-off between accuracy and speed. A precise subpixel disparity map can be estimated with a near real-time performance. Compared to conventional MRF-based algorithms, the proposed global refinement method in this chapter only aggregates the costs around the best disparity and updates the disparity map in a more efficient way. The proposed disparity refinement algorithm is developed based on the following assumptions:

- the subpixel disparity map obtained in section 5.2.2 is acceptable.

- for an arbitrary pixel, its neighbours (excluding discontinuities) in all directions have similar disparities.

- the interpolated parabola $f(d) = \beta_0 + \beta_1 d + \beta_2 d^2$ in Section 5.2.2.2 is locally smooth.

Before going into further details about the proposed disparity refinement approach, the author first rewrites the energy function in Eq. 2.42 in a more general way as follows [106]:

$$E(\boldsymbol{p}) = E_{data}(\boldsymbol{p_{ij}}) + \lambda E_{smooth}(\boldsymbol{p_{ij}}, \boldsymbol{n_{p_{ij}}}) \tag{5.3}$$

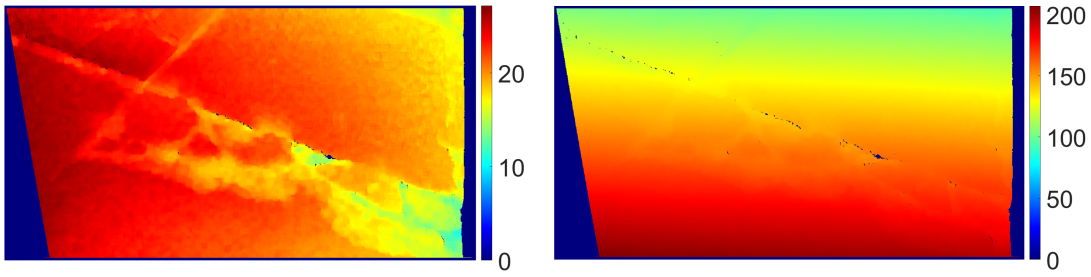
For conventional MRF-based stereo matching algorithms, E_{data} denotes the matching cost and E_{smooth} is the cost aggregation from the neighbourhood system. By minimising the global energy of the whole random field, a disparity map can be estimated.

In Section 5.2.2.2, a parabola $f(d) = \beta_0 + \beta_1 d + \beta_2 d^2$ is fitted to three correlation costs $c_{NCC}(u, v, d-1)$, $c_{NCC}(u, v, d)$ and $c_{NCC}(u, v, d+1)$ to get the subpixel disparity d_s. The parabola function $f(d)$ contains the information of both subpixel disparity and correlation costs. Since $f(d)$ is assumed to be locally smooth, the neighbouring pixels tend to have similar parabola parameters. However, when an abrupt change occurs, they vary significantly and in this case, the condition for uniform smoothness is no longer valid. Therefore, the function $f(d_{\boldsymbol{p_{ij}}})$ is used as the label in MRF. By adaptively aggregating functions $f(d_{\boldsymbol{n_{p_{ij}}}})$ of the neighbourhood system to $f(d_{\boldsymbol{p_{ij}}})$, $f(d_{\boldsymbol{p_{ij}}})$ is updated iteratively.

In order to ensure energy minimisation rather than energy maximisation as widely presented in literature, the term E_{data} is defined as:

$$E_{data}(\boldsymbol{p_{ij}}) = -f(d_{\boldsymbol{p_{ij}}}) \tag{5.4}$$

λ has a value of $1/\sqrt{2}$ in this chapter. Using the same strategy of adaptive aggregation in FBS, the author defines the smoothness energy $E_{smooth}(\boldsymbol{p_{ij}}, \boldsymbol{n_{p_{ij}}})$ as the adaptive sum of negative interpolated parabolas $-f(d_{\boldsymbol{n_{p_{ij}}}})$ of spatially varying horizontal and vertical nearest neighbours:



(a) (b)

Figure 5.5: Disparity map global refinement and post-processing. (a) subpixel disparity map after the third iteration. (b) post-processed disparity map.

$$E_{smooth}(\boldsymbol{p_{ij}}, \boldsymbol{n_{p_{ij}}}) = -\sum_{m=1}^{k} \omega(\boldsymbol{p_{ij}}, \boldsymbol{n_{mp_{ij}}}) f(d_{\boldsymbol{n_{mp_{ij}}}}) \tag{5.5}$$

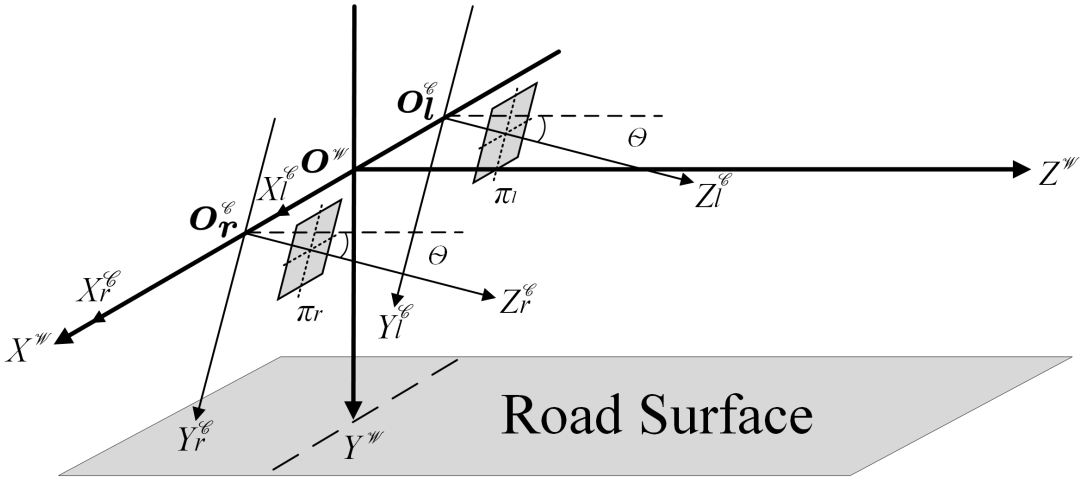
where

$$\omega(\boldsymbol{p_{ij}}, \boldsymbol{n_{mp_{ij}}}) = \exp\left\{-\frac{||\mathscr{E}_m||_2^2}{{\sigma_d}^2}\right\} \exp\left\{-\frac{(d_{\boldsymbol{n_{mp_{ij}}}} - d_{\boldsymbol{p_{ij}}})^2}{{\sigma_r}^2}\right\} \tag{5.6}$$

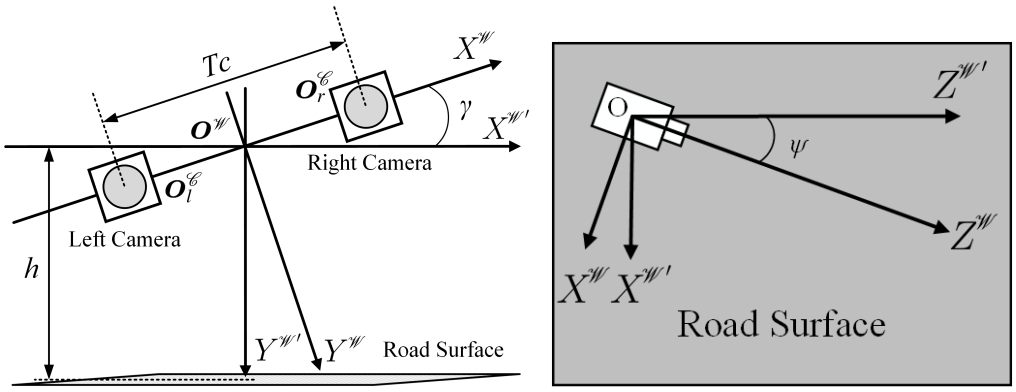
The weighting coefficient ω is determined by both the spatial distance $||\mathscr{E}_m||_2$ between $\boldsymbol{n_{mp_{ij}}}$ and $\boldsymbol{p_{ij}}$ and the difference between $d_{\boldsymbol{n_{mp_{ij}}}}$ and $d_{\boldsymbol{p_{ij}}}$. σ_d and σ_r are two parameters used to control ω and they are respectively set to 1 and 5 in this chapter. If $d_{\boldsymbol{n_{mp_{ij}}}}$ is similar to $d_{\boldsymbol{p_{ij}}}$, the weight for cost aggregation is higher. The energy function with respect to the correlation costs is updated iteratively. The subpixel disparity map is optimised by approximating the minima of the updated energy functions. In this chapter, the proposed process is iterated three times, and the result after the third iteration is shown in Figure 5.5a.

5.2.4 Post-Processing and 3-D reconstruction

Due to the fact that the perspective views have been transformed in subsection 5.2.1, the estimated subpixel disparities on row v should be added $\alpha_0 + \alpha_1 v - \delta$ to obtain the post-processed disparity map which is illustrated in Figure 5.5b. Then, the intrinsic and extrinsic parameters of the stereo system are used to compute each 3-D point $\boldsymbol{P^W} = [X^W, Y^W, Z^W]^\top$ from its projections $\boldsymbol{p_l} = [u_l, v_l]^\top$ and $\boldsymbol{p_r} = [u_r, v_r]^\top$, where v_r is equivalent to v_l, and u_r is associated with u_l by



(a)



(b) (c)

Figure 5.6: Extrinsic rotations. (a) pitch angle θ. (b) roll angle γ. (c) yaw angle ψ. h is the height of the proposed binocular system.

disparity d.

For many state-of-the-art road model estimation algorithms, the effects caused by the non-zero roll angle (Figure 5.6b) are always ignored because the stereo cameras will not change significantly over time [6]. However, the experimental set-up in this chapter is installed manually and the roll angle may introduce a distortion on the v-disparity histogram. Therefore, the roll angle needs to be estimated for the initial frame to minimise its impact on the perspective transformation for the rest of the sequences. As in [6], the roll angle γ can be estimated by fitting a linear plane ($d(u, v) = \gamma_0 + \gamma_1 u + \gamma_2 v$) to a small patch from the near field in

the disparity map and $\gamma = \arctan(-\gamma_1/\gamma_2)$. The pitch angle θ can be estimated by rearranging Eq. 5.1 as Eq. 5.7, where the parameters $[\alpha_0, \alpha_1]^\top$ have been approximated in section 5.2.1. The yaw angle ψ shown in Figure 5.6c is assumed to be 0.

$$\theta = \arctan\left(\frac{1}{f}\left(\frac{\alpha_0}{\alpha_1} + v_0\right)\right) \tag{5.7}$$

Each 3-D point $[X^{\mathscr{W}}, Y^{\mathscr{W}}, Z^{\mathscr{W}}]^\top$ can be transformed into $[X^{\mathscr{W}\prime}, Y^{\mathscr{W}\prime}, Z^{\mathscr{W}\prime}]^\top$ using Eq. 5.8 [107]. The rotation matrix $\boldsymbol{R} = \boldsymbol{R_\psi R_\theta R_\gamma}$ is a SO(3) matrix. The rotation with $\boldsymbol{R}$ makes pothole detection much easier. The 3-D reconstruction of Figure 5.3a is illustrated in Figure 5.7.

$$\begin{bmatrix} X^{\mathscr{W}\prime} \\ Y^{\mathscr{W}\prime} \\ Z^{\mathscr{W}\prime} \end{bmatrix} = \boldsymbol{R_\psi R_\theta R_\gamma} \begin{bmatrix} X^{\mathscr{W}} \\ Y^{\mathscr{W}} \\ Z^{\mathscr{W}} \end{bmatrix} \tag{5.8}$$

where

$$\boldsymbol{R_\psi} = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \tag{5.9}$$

$$\boldsymbol{R_\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{5.10}$$

$$\boldsymbol{R_\gamma} = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.11}$$

5.3 Experimental Results

In this section, the performance of the proposed road surface 3-D reconstruction algorithm is evaluated both qualitatively and quantitatively. The algorithm is programmed in C language on an Intel Core i7-4720HQ CPU using a single thread.

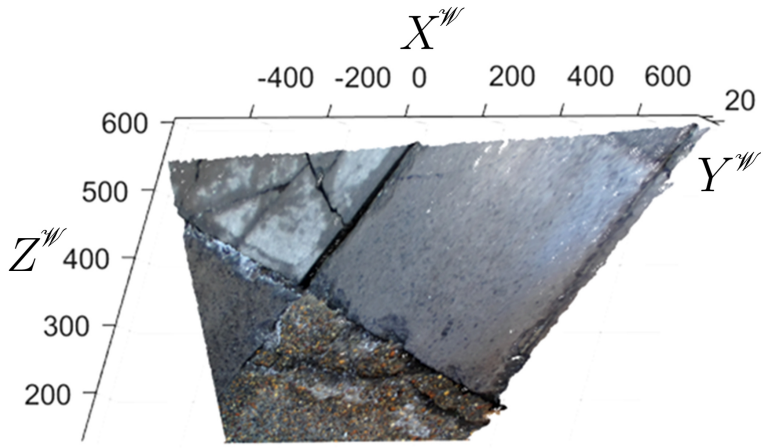


Figure 5.7: Road surface 3-D reconstruction.

The following subsections detail the experimental set-up and the performance evaluation.

5.3.1 Experimental Set-Up

In the experiments, a state-of-the-art stereo camera from ZED Stereolabs is used to capture 1080p (3840 × 1080) videos at 30 fps or 2.2K (4416 × 1242) videos at 15 fps [108]. The baseline is 120 mm. With its ultra sharp six element all-glass dual lenses and 16:9 native sensors, the video is 110° wide-angle and able to cover the scene up to 20 m. An example of the experimental set-up is shown in Figure

Figure 5.8: Experimental set-up.

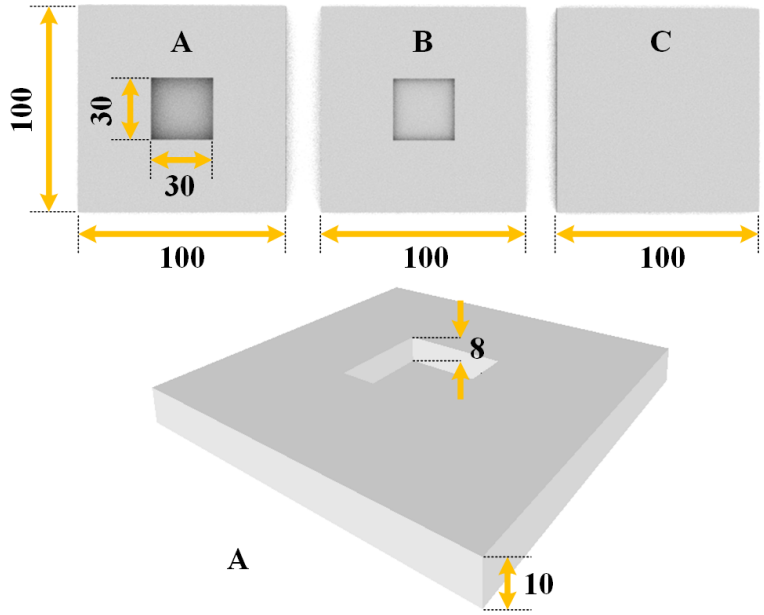


Figure 5.9: Designed 3-D sample models. The unit is millimetre.

5.8. The stereo camera is calibrated manually using the stereo calibration toolbox from MATLAB R2017a. The overall calibration mean error in pixels is 0.335.

To quantify the accuracy of the proposed algorithm, the author designed three sample models A, B and C with different sizes. They are printed with a MakerBot Replicator 2 Desktop 3-D Printer whose layer resolution is from 0.1 mm to 0.3 mm. Their top views and the stereogram of model A are illustrated in Figure 5.9, where A and B are designed with grooves to simulate potholes. To get the ground truth for the experiments, the author measured the actual size of these models using an electronic vernier caliper. Both the design and actual sizes of the models are presented in Table 5.1. Since the models are printed with a single colour, resulting in homogeneous areas, the author attached them with a piece of paper with the texture of the road surface printed on it to avoid the ambiguities during stereo matching, as can be seen in Figure 5.8.

Using the above experimental set-up, three datasets (91 stereo image pairs) are created for the road surface 3-D reconstruction. Datasets 1 and 2 aim at road sceneries, and dataset 3 contains the sample models to help researchers qualify their reconstruction results. The datasets are available at: http://

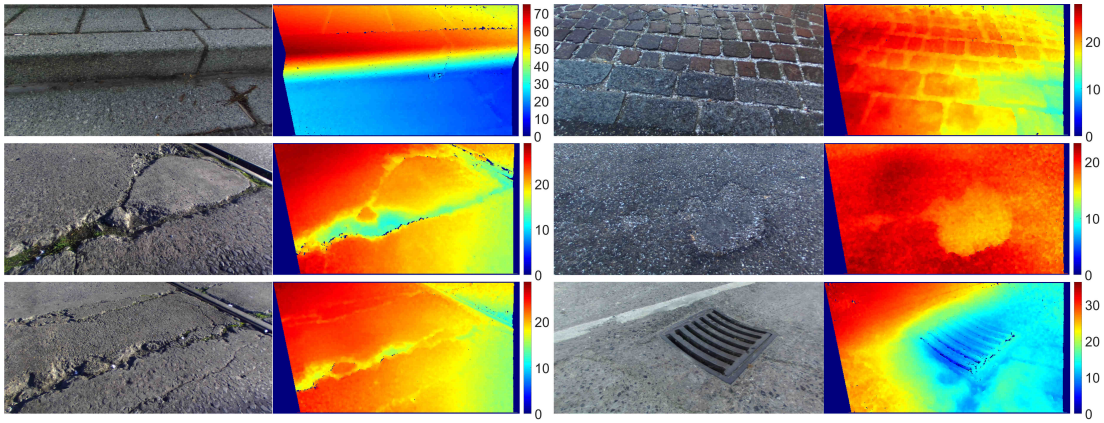


Figure 5.10: Experimental results. The first and third columns are the input left images. The second and fourth columns are the subpixel disparity map without post-processing.

Table 5.1: Design size and actual size of the sample models.

Sample model	Design size (mm×mm×mm)	
	Model	Groove
A	100.00 × 100.00 × 10.00	30.00 × 30.00 × 8.00
B	100.00 × 100.00 × 10.00	30.00 × 30.00 × 3.00
C	100.00 × 100.00 × 5.00	n/a
Sample model	Actual size (mm×mm×mm)	
	Model	Groove
A	99.97 × 99.83 × 10.31	29.74 × 30.01 × 8.25
B	100.39 × 100.10 × 9.82	30.28 × 29.98 × 3.52
C	100.00 × 99.98 × 5.92	n/a

www.ruirangerfan.com.

The following subsections analyse the performance of the proposed algorithm in terms of disparity accuracy, reconstruction accuracy and processing speed.

5.3.2 Disparity Evaluation

Some examples of the disparity maps are illustrated in Figure 5.10. Before estimating the disparity map, the target image is transformed into its reference view, which greatly eliminates the perspective distortion for a GP between two images. Since the GP in the left and right images now looks similar to each other, the average of the highest correlation costs goes higher, which is depicted in Figure

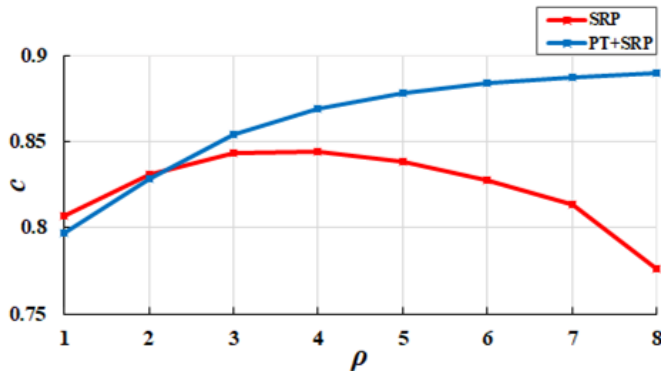


Figure 5.11: Comparison between SRP and PT+SRP in terms of the average of the highest correlation costs.

5.11. For stereo matching with only SRP, the average of the highest correlation increases gradually from 0.807 ($\rho = 1$) to 0.845 ($\rho = 4$). However, when ρ goes above 4, c keeps decreasing. If the input image pairs are pre-processed with the PT, the average of the highest correlation costs in the SRP stereo will grow gradually between $\rho = 1$ and $\rho = 8$. In this chapter, the datasets are created with high-resolution images, and ρ is proposed to be 5. Compared with the conventional SRP stereo, the PT improves the average correlation cost with an increase of 0.05.

Furthermore, the author selects one row from the disparity map to evaluate the performance of subpixel enhancement and global refinement (see Figure 5.12). The integer disparity d oscillates along the selected row and drops down abruptly when a discontinuity occurs. After the subpixel enhancement, the disparity d

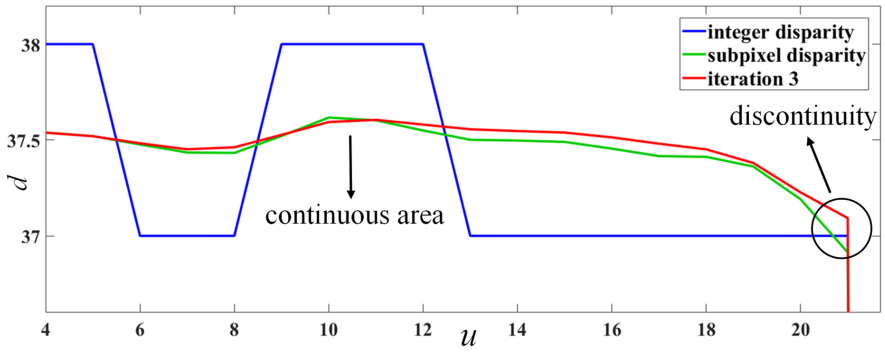


Figure 5.12: Evaluation of subpixel enhancement and disparity global refinement.

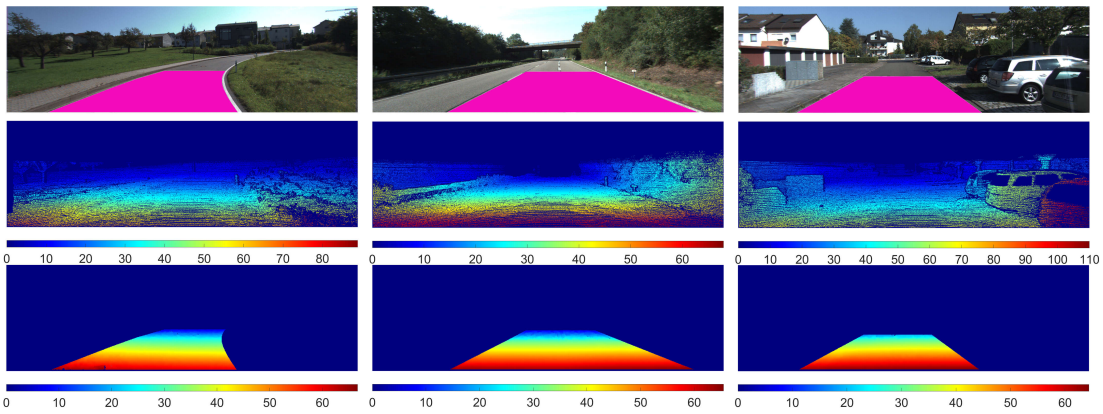


Figure 5.13: Experimental results of the KITTI stereo 2012 dataset. The first row shows the left images, where areas in magenta are the manually selected road surface. The second row shows the disparity ground truth. The third row shows the results obtained from the proposed algorithm.

is replaced with a better one d_s between $d-1$ and $d+1$. The iterative global refinement further optimises the subpixel disparity map. After the third iteration, the disparities change more smoothly in a continuous area but interrupt suddenly when reaching a discontinuity.

Since the created datasets only contain the ground truth of 3-D reconstruction, the KITTI stereo 2012 dataset [5] is used to further evaluate the disparity accuracy of the proposed algorithm. Some experimental results are illustrated in Figure 5.13. Due to the fact that the proposed algorithm only aims at reconstructing the road surface, the author selects a region of interest (see the magenta areas in the first row) from each image to evaluate the performance of the proposed algorithm. The corresponding disparity results in the region of interest are shown in the third row. The percentage of error pixels (threshold: two pixels) is around 0.73% and the average error in pixels is about 0.51.

5.3.3 Reconstruction Evaluation

To further evaluate the accuracy of the reconstruction results, the author created dataset 3 (see section 5.3.1 for details) with three different sample models. An example of the left image is illustrated in Figure 5.14a. The corresponding subpixel disparity map and 3-D reconstruction are depicted in Figure 5.14b and

Figure 5.14c, respectively. The author selects a rectangular region which includes one of the sample models from Figure 5.14a, and the 3-D reconstruction of this region can be seen in Fig. 5.14d. A surface $\kappa_0 X^{\mathscr{W}} + \kappa_1 Y^{\mathscr{W}} + \kappa_2 Z^{\mathscr{W}} + \kappa_3 = 0$ is fitted to four corners $\boldsymbol{s}_1^{\mathscr{W}}$, $\boldsymbol{s}_2^{\mathscr{W}}$, $\boldsymbol{s}_3^{\mathscr{W}}$ and $\boldsymbol{s}_4^{\mathscr{W}}$ of the selected region. Then, the author selects a set of random points $\boldsymbol{p}_1^{\mathscr{W}}, \boldsymbol{p}_2^{\mathscr{W}}, \ldots, \boldsymbol{p}_n^{\mathscr{W}}$ on the surface of the model and estimate the distances between them and the fitted road surface. These random distances provide the measurement range of the model height. Similarly, the groove depth can be estimated by computing the distances between a group of points $\boldsymbol{q}_1^{\mathscr{W}}, \boldsymbol{q}_2^{\mathscr{W}}, \ldots, \boldsymbol{q}_n^{\mathscr{W}}$ in a groove and the model surface. Table 5.2 details the range of the measured model height and groove depth, where D represents the approximated distance from the camera to sample models.

From Table 5.2, the maximal absolute error of the 3-D reconstruction is ap-

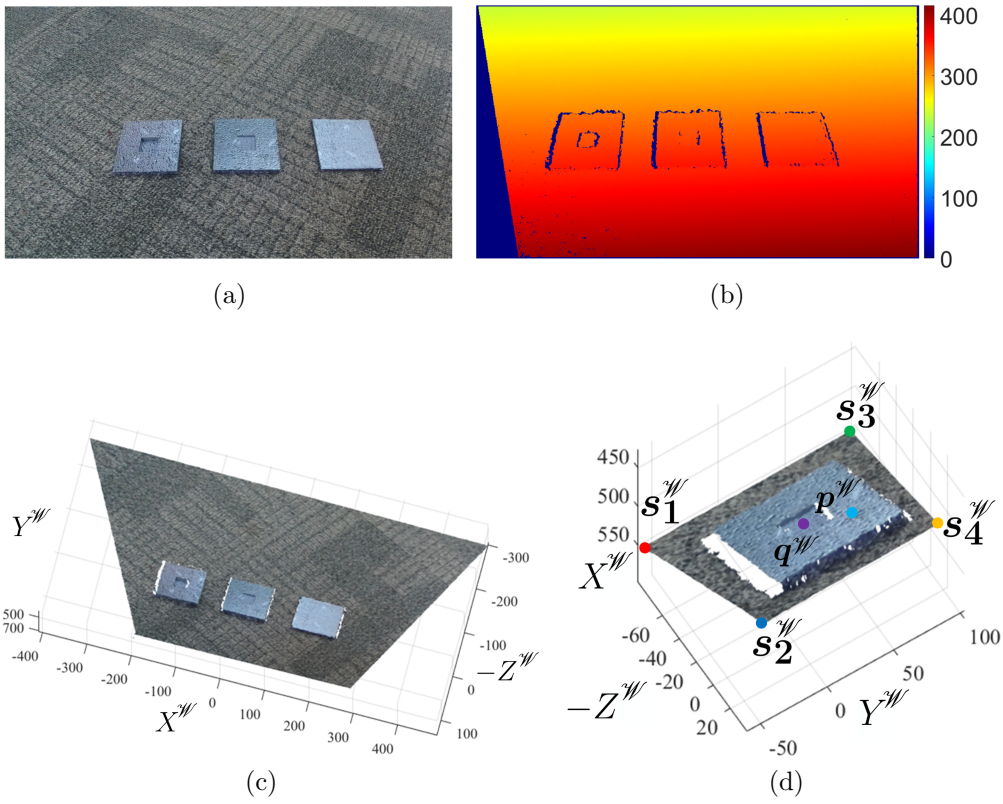


Figure 5.14: Sample model 3-D reconstruction. (a) left image. (b) subpixel disparity map with post-processing. (c) reconstructed scenery. (d) selected 3-D point cloud which includes model B.

Table 5.2: 3-D reconstruction measurement range.

Target	Measurement range (mm)				
	$D \approx 450mm$	$D \approx 470mm$	$D \approx 500mm$	$D \approx 550mm$	$D \approx 650mm$
A height	09.72 – 10.21	09.64 – 11.12	10.31 – 12.19	09.59 – 12.37	08.99 – 12.62
B height	09.86 – 10.32	09.91 – 10.47	10.07 – 11.25	10.10 – 11.99	10.86 – 12.36
C height	04.62 – 05.54	04.92 – 06.11	05.72 – 06.93	06.61 – 07.18	06.69 – 07.54
A grove	07.77 – 08.44	08.31 – 09.54	05.92 – 09.17	05.49 – 07.26	09.37 – 11.83
B grove	02.21 – 05.12	04.88 – 05.32	04.97 – 06.51	06.28 – 07.57	05.29 – 06.63

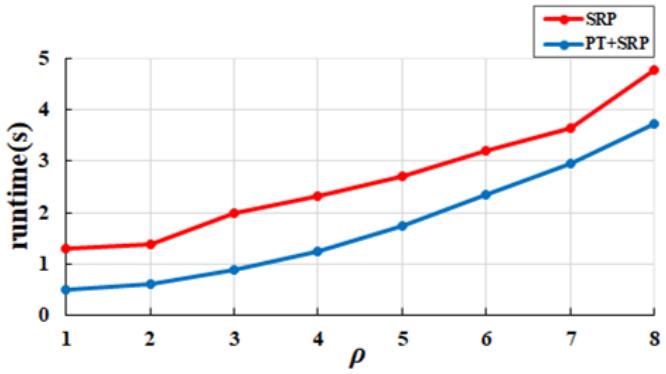


Figure 5.15: Comparison between SRP and PT+SRP in terms of the runtime.

proximately 3 mm, and it increases slightly when D increases. The reconstruction precision is inversely proportional to the depth [109]. Furthermore, since the baseline of the ZED camera is fixed and cannot be increased to further improve the precision, it is mounted to a relatively low height and kept as perpendicular as possible to the road surface to reduce the average depth, which guarantees a high reconstruction accuracy.

5.3.4 Processing Speed

The algorithm is implemented in C language on an Intel Core i7-4720HQ CPU (2.6 GHz) using a single thread. After the PT, each point on row v in the target image is shifted $a_0 + a_1 v - \delta$ pixels to obtain a reference view, which greatly reduces the search range for stereo matching. The evaluation of the PT with respect to the runtime is illustrated in Figure 5.15. The PT accelerates the processing speed of the SRP stereo when using different block sizes. When $\rho = 5$, the processing speed is increased by over 36%. The runtime of different datasets

is shown in Table 5.3. Although the proposed algorithm does not run in real time, the author believes that its speed can be increased in the future by exploiting the parallel computing architectures.

Table 5.3: Algorithm runtime.

Dataset	Frames	Resolution	Runtime (s)
Dataset 1	35	1240×609	0.71
Dataset 2	35	1249×620	0.84
Dataset 3	21	2081×1048	2.23

5.4 Conclusion

The main novelties of this chapter include PT, CMV, and disparity map global refinement. The author created three datasets and made them publicly available to contribute to 3-D reconstruction-based pothole detection. The PT not only enhances the similarity of a GP between two images but also reduces the search range for stereo matching. This helps the SRP stereo perform more accurately and efficiently. The CMV further offsets the insufficient propagation in the SRP stereo and guarantees the feasibility of parabola interpolation in the subpixel enhancement phase. By iteratively minimising the energy with respect to the interpolated parabolas, the subpixel disparity map is optimised. The disparities in a continuous area become more smooth, but they are preserved when discontinuities occur. The maximal absolute error of the 3-D reconstruction is around 3 mm, which satisfies the requirement of millimetre accuracy for on-road damage detection.

Chapter 6

Robust Pothole Detection, Classification and Tracking System Based on Computer Stereo Vision Technique

Detecting potholes is one of the most important tasks in pavement condition assessment. The pothole detection approaches based on computer vision technology can mainly be classified as 2-D image processing-based and 3-D modelling-based. However, these approaches are usually used independently and the detection accuracy is always unsatisfactory. Hence this chapter presents a robust pothole detection, classification and tracking system based on stereo vision technique. The disparity map obtained from the stereo vision is first transformed to better distinguish the potholes from the road surface and a segmentation performed on the transformed disparity map can therefore separate distress and non-distress areas accurately. To achieve a better processing efficiency of the disparity map transformation, Golden Section Search (GSS) and DP are utilised to estimate the transformation parameters efficiently. Then, a robust two-step disparity map modelling algorithm is proposed to fit a quadratic surface to the disparities in the non-distress area. The surface coefficients are coarsely estimated in the first step using the LSF and RANSAC. In the second step, the surface coefficients

are updated iteratively to refine the precision. The gradient information is also integrated into the process of surface fitting when determining the inliers and outliers in each iteration. The different potholes are classified using the CCL and the same pothole in successive frames is tracked using the Discriminative Scale Space Tracking (DSST). The experimental results illustrate that the successful detection rate of the proposed system is approximately 99%.

6.0.1 Motivations

Currently, the laser scanning is the main technology that is used to provide 3-D information for pothole detection, while other technologies such as passive sensing are underutilised [110]. However, the laser scanning equipment mounted on DIVs is still costly and cannot be adapted for other vehicles. Recently, with some major advancements have been achieved in stereo vision research field regarding the estimation of subpixel disparity maps, the binocular system described in Chapter 5 can reconstruct 3-D road surface with an accuracy of three millimetres [40]. Also, the stereo cameras used for road condition assessment are inexpensive, portable and adapted for different types of vehicles. Therefore, one of the motivations of this chapter is to explore the possibility of using stereo vision for pothole detection.

Furthermore, as discussed in Chapter 2, comprehensive studies have been made in the area of 2-D image processing and 3-D modelling for pothole detection. However, these algorithms are usually used independently and the accuracy of the modelled road surface is severely affected by the outliers used for fitting [81]. Therefore, another motivation of this chapter is to explore an efficient way of segmenting the 3-D information, e.g., disparity map, with commonly used 2-D image processing algorithms. Then, only the candidates in a non-distress area are used for surface fitting.

Moreover, the gradient information is rarely integrated into surface fitting in the existing 3-D modelling-based pothole detection algorithms. Hence this chapter improves on the works performed in [88] and [111] to enhance the robustness of surface modelling by eliminating the candidates whose gradients differ significantly from the expected ones.

Finally, no tracking algorithms are integrated into the existing pothole detection and classification systems. Therefore, in this chapter the author uses DSST for the purpose of pothole tracking.

6.0.2 Contributions

In this chapter, a robust pothole detection, classification and tracking system is presented. The main contributions include a novel disparity map transformation algorithm and a two-step disparity map modelling algorithm.

To fit a quadratic surface to the dense disparity map, the distress and non-distress areas should be discriminated [84]. Therefore, the disparity map is first transformed to better distinguish the potholes from the road surface. To achieve a better processing efficiency of the disparity map transformation, GSS and DP are used to estimate the transformation parameters. A segmentation performed on the transformed disparity map can thus separate the distress and non-distress areas accurately. Then, a novel two-step disparity map modelling algorithm is carried out to fit the input disparity map into a quadratic surface for pothole detection. In the first step, the RANSAC is jointly used with the LSF to roughly estimate the parameters of the quadratic surface, where the difference of both disparity values and gradient orientations between the actual and the modelled disparity maps are used to determine the inliers and outliers for the RANSAC. In the second step, the surface parameters and the points used for surface modelling are updated iteratively until the number of outliers becomes 0, which greatly improves the accuracy of fitted quadratic surface. Finally, the detected potholes are tracked using the DSST.

The author also created three synthetic pothole datasets using a ZED stereo camera. The datasets (containing RGB images, disparity maps and 3-D point clouds) are publicly available at: http://www.ruirangerfan.com. The dataset presented in [87] is used for pothole tracking. More details on the datasets and the experimental set-up can be seen in section 6.2.

The rest of the chapter is structured as follows: Section 6.1 details the proposed pothole detection system. The experimental results are illustrated in Section 6.2 and the performance of the algorithm is evaluated. Finally, Section 6.3

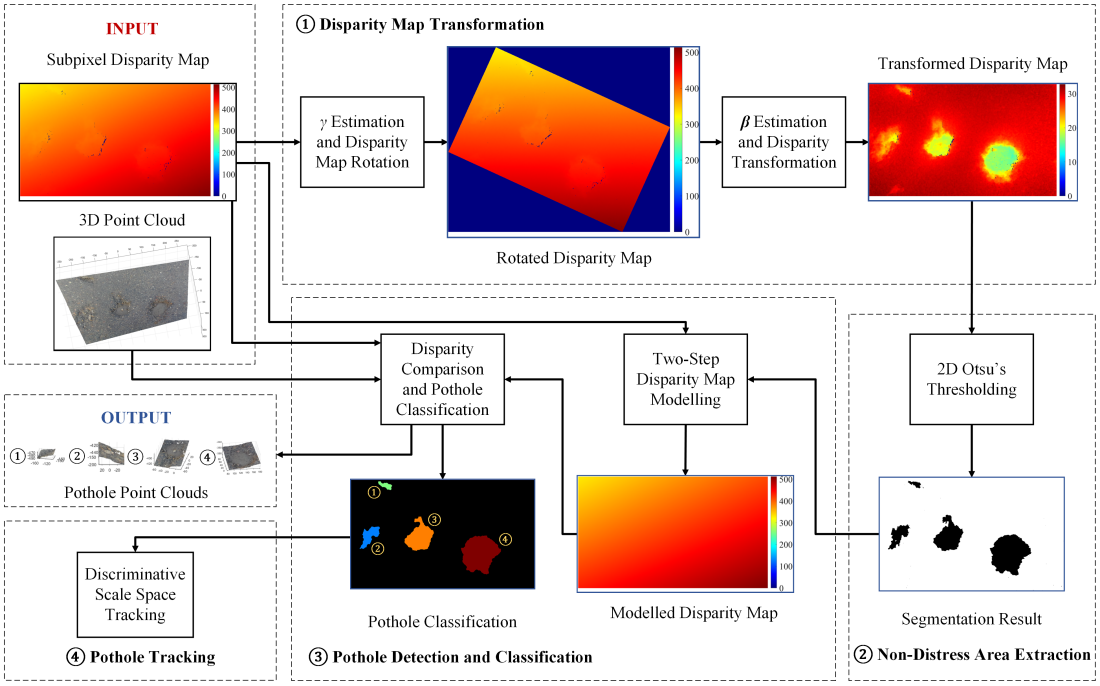


Figure 6.1: The overview of the proposed pothole detection, classification and tracking system.

summarises the chapter.

6.1 Algorithm Description

The overview of the proposed pothole detection, classification and tracking system is illustrated in Figure 6.1. The proposed system consists of four main procedures: disparity map transformation, non-distress area extraction, pothole detection and classification and pothole tracking.

The input of the proposed system is a dense subpixel disparity map and its corresponding 3-D point cloud data which are obtained using the 3-D reconstruction algorithm presented in Chapter 5. Firstly, a disparity map transformation algorithm is carried out to better distinguish the potholes from the road surface. This is achieved by using two transformation parameters γ and $\boldsymbol{\beta}$, where γ denotes the roll angle (see Figure 5.6b) and $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2]^\top$ is a vector storing the coefficients of the vertical road pattern. More details on $\boldsymbol{\beta}$ are provided in

Chapter 4. After the disparity map transformation, the disparity values of a non-distress area become uniform while they differ greatly from those of a distress area. This greatly improves the extraction of the non-distress area. The latter provides a region of interest where the points are used for disparity map modelling. To improve the accuracy of the fitted quadratic surface, the author proposes a two-step disparity map modelling algorithm, where the surface parameters $\boldsymbol{c} = [c_0, c_1, c_2, c_3, c_4, c_5]^\top$ are estimated roughly in the first step using the LSF and the RANSAC and their values are updated iteratively in the second step for a finer estimate. Furthermore, the gradient of each point is also integrated into the surface fitting to determine the inliers and outliers for the RANSAC. By comparing the difference between the actual and modelled disparity values of each point, the potential pothole areas can be extracted. After a post-processing which eliminates the small objects and fills the small holes, different potholes are classified using the CCL. Finally, the spatial structures of the detected potholes are illustrated by extracting the corresponding regions from the input 3-D point cloud data. The same pothole in successive frames are tracked using the DSST. The rest of this section details these four procedures. The experimental results will be discussed in section 6.2.

6.1.1 Disparity Map Transformation

The proposed disparity map transformation algorithm is composed of two main steps: γ estimation and disparity map rotation, $\boldsymbol{\beta}$ estimation and disparity transformation.

Over the past two decades, a lot of research has been carried out to estimate γ using different technologies, such as Inertial Measurement Units (IMU) [112–115] and passive sensing [116]. The methods based on IMU usually combine the data acquired using multiple sensors, e.g., GPS, accelerometers and gyroscopes, to estimate the orientation of a vehicle [114, 115]. This not only makes the set-up cost unreasonably high but also complicates the estimation procedure [116]. Furthermore, in these approaches the road bank angle is always considered to be zero and only the roll angle γ is considered in the estimation process. But in real-world applications, the road bank angle is not always zero and hence both angles have to

be estimated independently and this is not usually a straightforward process [113]. In recent years, some passive sensor-based methods [6, 40, 116–119] have been proposed to address this issue. However, most of the authors used a single camera to acquire the road data and made some inaccurate assumptions. For example, they considered constant lane markings width to ensure their algorithms work properly [118]. However, the roll angle usually changes over time in actual cases, and therefore such assumptions are not always valid. In this regard, some authors have resorted to 3-D information to estimate an accurate roll angle [6, 118, 119]. In [118] and [119], the author assumed that the road surface is a ground plane and estimated the roll and pitch angles from a so-called "v-disparity map". In [40] and [6], a plane $d(u, v) = \gamma_0 + \gamma_1 u + \gamma_2 v$ (where (u, v) is the coordinate of a pixel in the disparity map) is fitted to a small patch which is selected from the near field in the disparity map and $\gamma = \arctan(-\gamma_1/\gamma_2)$. However, the above stereo vision-based algorithms are only suitable for a flat road surface which can be assumed to be a ground plane. Furthermore, selecting a proper patch for plane fitting is challenging because it may contain an obstacle or a pothole, which can severely affect the accuracy of the plane fitting. Hence in this chapter, the road surface is assumed to be non-flat and its disparity projection on the v-disparity map is a parabola $f(v) = \beta_0 + \beta_1 v + \beta_2 v^2$. The author first proposes an efficient γ estimation algorithm based on the GSS. The input disparity map is then ro-

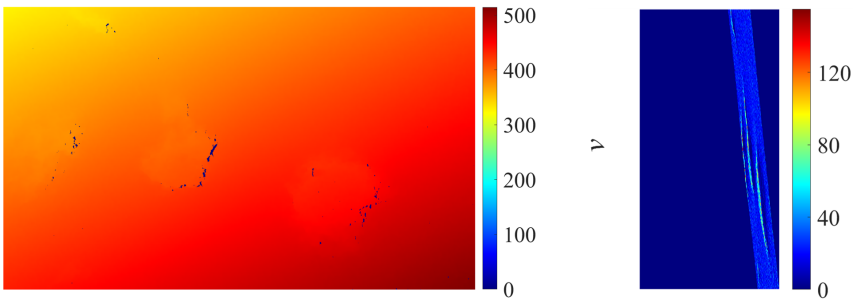


Figure 6.2: The input dense subpixel disparity map whose roll angle is non-zero and its corresponding v-disparity map. (a) input dense subpixel disparity map ($\gamma \neq 0$). (b) the v-disparity map of Figure 6.2a.

tated around it. This makes the disparity distribution of each row become more compact, which further improves the estimation of $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2]^\top$. The target projection on the v-disparity map is then extracted using Eq. 4.2. Finally, the rotated disparity map is transformed using the parameters $\boldsymbol{\beta}$ and γ to better distinguish the potential pothole areas from the road surface.

6.1.1.1 γ Estimation and Disparity Map Rotation

In this chapter, the input dense disparity map ℓ^{ipt} is estimated using the algorithm described in Chapter 5, and the parameters $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2]^\top$ can be estimated by solving the least squares problem in Eq. 4.3.

For a stereo rig which is ideally horizontal to the road surface, the roll angle of the stereo rig is zero. The disparity values for each row are similar while they change gradually in the vertical direction. However, a non-zero roll angle introduced from the set-up installation makes the disparities change gradually in each row (see Figure 6.2a), and therefore the disparity distribution of each row becomes less compact (see Figure 6.2b) compared to the case when γ is zero (see Figure 6.4b). This greatly affects the accuracy of the LSF, which makes the minimum energy E_{min} higher than the desired value. Therefore, the aim of the proposed roll angle estimation algorithm is to rotate ℓ^{ipt} at different angles and then find the angle at which the minimum E_{min} is obtained, as shown in Figure 6.3.

To rotate ℓ^{ipt} around a given angle γ, each coordinate (u, v) in the original disparity map is transformed to a new coordinate (u', v') using Eq. 6.1 and 6.2. where (u_o, v_o) is the coordinate of the centre of ℓ^{ipt}.

$$u' = (u - u_o)\cos\gamma + (v - v_o)\sin\gamma \tag{6.1}$$

$$v' = (v - v_o)\cos\gamma - (u - u_o)\sin\gamma \tag{6.2}$$

After the coordinate translation and rotation, the position (u', v') in the rotated disparity map ℓ^{rot} has the same disparity value as the position (u, v) in ℓ^{ipt}, and the energy function in Eq. 4.3 can thus be rewritten as Eq. 6.3.

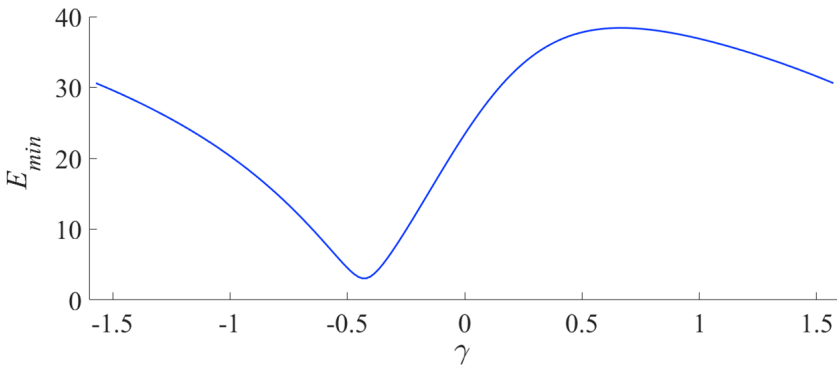


Figure 6.3: The relationship between the minimum energy E_{min} and different angles γ.

Algorithm 10: γ estimation using the GSS.

Data: ℓ^{ipt}
Result: γ
set γ_1 and γ_2 to $-\pi/2$ and $\pi/2$, respectively;
minimise E in Eq. 6.3 with respect to γ_1 and get E_{min1};
minimise E in Eq. 6.3 with respect to γ_2 and get E_{min2};
while $\gamma_2 - \gamma_1 > \kappa$ **do**
    set γ_3 and γ_4 to $k\gamma_1 + (1-k)\gamma_2$ and $k\gamma_2 + (1-k)\gamma_1$, respectively;
    minimise E in Eq. 6.3 with respect to γ_3 and get E_{min3};
    minimise E in Eq. 6.3 with respect to γ_4 and get E_{min4};
    if $E_{min3} > E_{min4}$ **then**
        γ_1 is replaced by γ_3;
    else
        γ_2 is replaced by γ_4;
    end
end

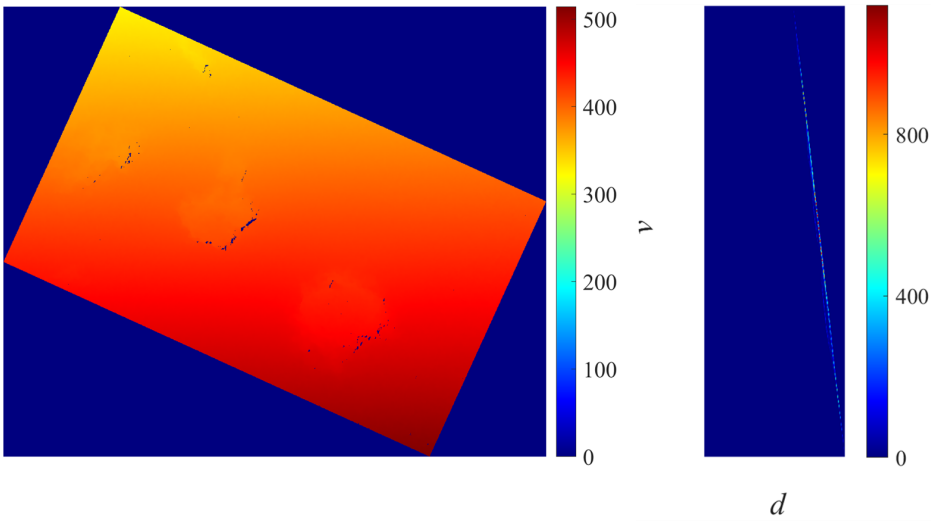
$$E = \sum_{i=0}^{n} \left(d_i - \left(\alpha_0 + \alpha_1 v_i' + \alpha_2 v_i'^2\right)\right)^2 \tag{6.3}$$

An arbitrary v' can be computed from u and v using Eq. 6.2 and the corresponding E_{min} is obtained by solving the energy minimisation problem in Eq. 4.3 using the LSF. It is to be noted that whether the disparity map is rotated around γ or $\gamma + \pi$, the same E_{min} will be obtained since $\cos(\gamma + \pi) = -\cos\gamma$ and $\sin(\gamma + \pi) = -\sin\gamma$. Therefore, the interval of γ is set to $(-\pi/2, \pi/2]$ and

find the desirable roll angle which is the position of the local minima as shown in Figure 6.3.

However, finding the local minima is a computationally intensive task because the whole interval has to be gone through. Furthermore, in order to obtain an accurate γ which corresponds to E_{min}, the step size κ should be set to a very small and practical value. Hence in this chapter, the GSS is utilised to reduce the search range within the interval $(-\pi/2, \pi/2]$. The procedures of the proposed γ estimation algorithm are given in algorithm 10, where $k = 0.618$ is the golden section factor. More details on the GSS are available in [120].

The dense disparity map is rotated around γ which is estimated using algorithm 10 and the corresponding result is illustrated in Figure 6.4a. Then, the v-disparity map (see Figure 6.4b) is created for the rotated dense disparity map and it can be observed that the disparity distribution of each row becomes more uniform. The performance evaluation of the proposed γ estimation algorithm will be discussed in Section 6.2.



(a) (b)

Figure 6.4: The rotated dense subpixel disparity map and its corresponding v-disparity map. (a) rotated dense disparity map ℓ^{rot} ($\gamma = 0$). (b) the v-disparity map of Figure 6.4a.

Algorithm 11: Disparity map transformation.

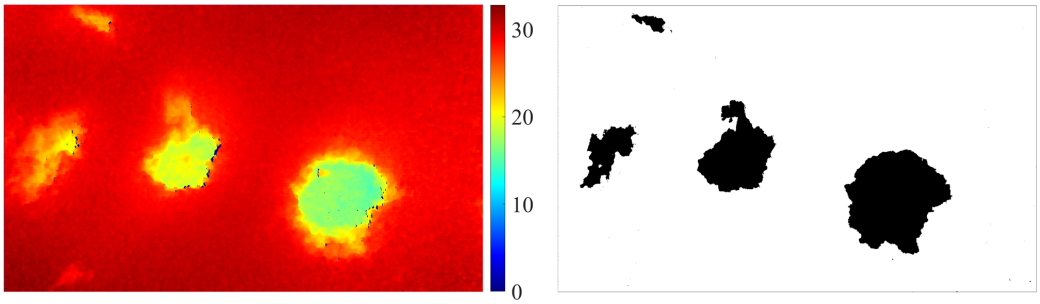
Input : ℓ^{rot}, γ and $\boldsymbol{M_v} = [\boldsymbol{d}, \boldsymbol{v}]^\top$
Output: ℓ^{trf}

estimate the coefficients of $f(v)$ from the v-disparity map;
set the disparity values in ℓ^{rot} to $d - f(v) + \delta$;
rotate the updated ℓ^{rot} around $-\gamma$ to get ℓ^{trf};

6.1.1.2 $\boldsymbol{\beta}$ Estimation and Disparity Transformation

To extract the target path from the v-disparity map, the DP searches for every possible solution using Eq. 4.2, where $m(d, v)$ is the disparity accumulation at the position of (d, v) on the v-disparity map and λ is the smoothness term. Then, the author selects $\boldsymbol{M_v} = [\boldsymbol{d}, \boldsymbol{v}]^\top$ with the minimal energy as the optimal solution, where $\boldsymbol{d} = [d_0, d_1, \cdots, d_m]^\top$ and $\boldsymbol{v} = [v_0, v_1, \cdots, v_m]^\top$ record the path of the optimal solution. The parameters of $\boldsymbol{\beta}$ can therefore be estimated using Eq. 4.3. More details are provided in Chapter 4.

The input dense disparity map (Figure 6.2a) can subsequently be transformed, as shown in Figure 6.5a. The disparities of the road surface area become uniform but they differ significantly from those of the pothole area as shown in Figure 6.5a. More details on the proposed disparity transformation are given in algorithm 11, where δ is set to 30 to ensure that the disparity values in the transformed disparity map ℓ^{trf} are always positive.



(a) (b)

Figure 6.5: Disparity map transformation and non-distress area extraction. (a) transformed disparity map. (b) segmentation result of Figure 6.5a.

6.1.2 Non-Distress Area Extraction

After the disparity transformation as explained in subsection 6.1.1, the distress areas become highly distinguishable from a non-distress area in the transformed disparity map, which greatly improves the performance of those histogram-based image segmentation methods. In this chapter, the author employs the two-dimensional Otsu's method to separate the distress and non-distress areas in Figure 6.5a. More details on Otsu's thresholding method are available in [121, 122]. The corresponding segmentation result of Figure 6.5a is illustrated in Figure 6.5b, where the distress and non-distress areas are shown in black and white, respectively. In subsection 6.1.3, only the pixels in the non-distress areas are used for disparity map fitting.

6.1.3 Pothole Detection and Classification

6.1.3.1 Two-Step Disparity Map Modelling

For 3-D modelling-based pothole detection approaches [87–89], the road surface is usually modelled as a quadratic surface and the pixels of a pothole can be extracted when the difference between their actual positions and the interpolated quadratic model exceeds a pre-set threshold ε_d.

In [87], Zhang et al. fitted the quadratic model $Y^{\prime\prime\prime} = c_0 + c_1 X^{\prime\prime\prime} + c_2 Z^{\prime\prime\prime} + c_3 X^{\prime\prime\prime 2} + c_4 X^{\prime\prime\prime} Z^{\prime\prime\prime} + c_5 Z^{\prime\prime\prime 2}$ to n points $[X^{\prime\prime\prime}, Y^{\prime\prime\prime}, Z^{\prime\prime\prime}]^\top$ in the WCS. The parameters $\boldsymbol{c} = [c_0, c_1, c_2, c_3, c_4, c_5]^\top$ can be obtained by solving the function in Eq. 6.4, where $\boldsymbol{M}$ is a Vandermonde matrix and $\boldsymbol{y}$ is a column vector recording the values of $Y^{\prime\prime\prime}$. By comparing the difference $\Delta Y^{\prime\prime\prime}$ between each pair of points on the actual and fitted road surface, the potential pothole areas can be extracted.

$$\boldsymbol{M}^\top \boldsymbol{M} \boldsymbol{c} = \boldsymbol{M}^\top \boldsymbol{y} \tag{6.4}$$

where

$$\boldsymbol{M} = \begin{bmatrix} 1 & X_1^{\prime\prime\prime} & Z_1^{\prime\prime\prime} & X_1^{\prime\prime\prime 2} & X_1^{\prime\prime\prime} Z_1^{\prime\prime\prime} & Z_1^{\prime\prime\prime 2} \\ 1 & X_2^{\prime\prime\prime} & Z_2^{\prime\prime\prime} & X_2^{\prime\prime\prime 2} & X_2^{\prime\prime\prime} Z_2^{\prime\prime\prime} & Z_2^{\prime\prime\prime 2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_n^{\prime\prime\prime} & Z_n^{\prime\prime\prime} & X_n^{\prime\prime\prime 2} & X_n^{\prime\prime\prime} Z_n^{\prime\prime\prime} & Z_n{}^2 \end{bmatrix} \tag{6.5}$$

$$\boldsymbol{y} = [Y_1^{\prime\prime\prime}, Y_2^{\prime\prime\prime}, \ldots, Y_n^{\prime\prime\prime}]^\top \tag{6.6}$$

Later on, Ozgunalp et al. integrated the gradient $\boldsymbol{g} = [g_u, g_v]^\top$ information into the surface fitting. Eq. 6.4 is therefore developed as Eq. 6.7 [88], where λ is the weighting for the gradient term. The matrices $\boldsymbol{N_u}$ and $\boldsymbol{N_v}$ are derived from the computation of the horizontal and vertical gradients using the parameters of $\boldsymbol{c}$. The column vectors $\boldsymbol{g_u}$ and $\boldsymbol{g_v}$ contain the horizontal and vertical derivatives, respectively.

$$\big(\boldsymbol{M}^\top \boldsymbol{M} + \lambda\big(\boldsymbol{N_u}^\top \boldsymbol{N_u} + \boldsymbol{N_v}^\top \boldsymbol{N_v}\big)\big)\boldsymbol{c} = \boldsymbol{M}^\top \boldsymbol{y} + \lambda\big(\boldsymbol{N_u}^\top \boldsymbol{g_u} + \boldsymbol{N_v}^\top \boldsymbol{g_v}\big) \tag{6.7}$$

where

$$\boldsymbol{N_u} = \begin{bmatrix} 0 & 1 & 0 & 2X_1^{\prime\prime\prime} & Z_1^{\prime\prime\prime} & 0 \\ 0 & 1 & 0 & 2X_2^{\prime\prime\prime} & Z_2^{\prime\prime\prime} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & 2X_n^{\prime\prime\prime} & Z_n^{\prime\prime\prime} & 0 \end{bmatrix} \tag{6.8}$$

$$\boldsymbol{N_v} = \begin{bmatrix} 0 & 0 & 1 & 0 & X_1^{\prime\prime\prime} & 2Z_1^{\prime\prime\prime} \\ 0 & 0 & 1 & 0 & X_2^{\prime\prime\prime} & 2Z_2^{\prime\prime\prime} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & 0 & X_n^{\prime\prime\prime} & 2Z_n^{\prime\prime\prime} \end{bmatrix} \tag{6.9}$$

$$\boldsymbol{g_u} = [g_{u_1}, g_{u_2}, \ldots, g_{u_n}]^\top \tag{6.10}$$

$$\boldsymbol{g_v} = [g_{v_1}, g_{v_2}, \ldots, g_{v_n}]^\top \tag{6.11}$$

As shown in Figure 6.6, the gradients of the pixels in a non-distress area usually differ greatly from those at the boundary of a distress area, but they are almost same as those in a distress area (excluding the boundary). Therefore, the algorithm in [88] only increases the weighting of the pothole boundary for surface fitting and the modelling accuracy is still affected by the interior part of the potholes. Furthermore, determining the optimum λ is not a straightforward process because it cannot be derived mathematically and therefore several experiments have to be carried out to get the value for λ. But even then, each frame might yield a different optimum λ. Also, in some cases $\lambda = 0$ when the

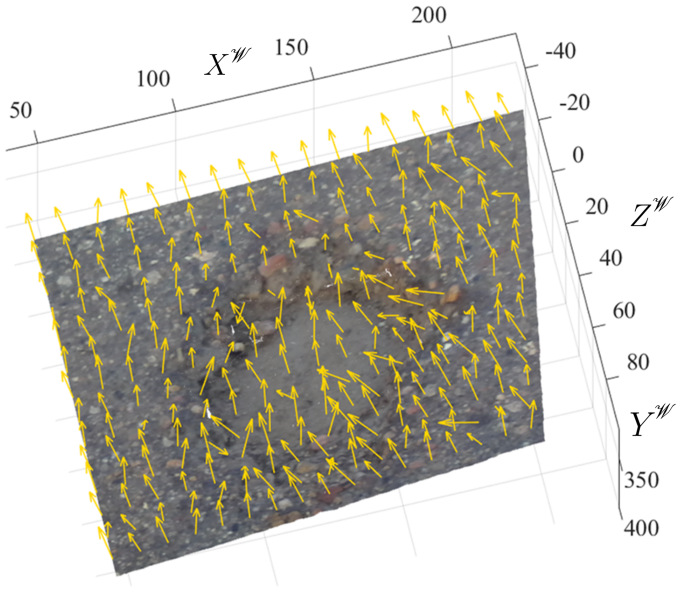


Figure 6.6: The road surface 3-D point cloud and the surface normal.

algorithm achieves the best modelling accuracy and therefore the optimisation solution in Eq. 6.7 is not always valid. Moreover, the values of X^W, Y^W and Z^W are inversely proportional to the corresponding disparity value d, and therefore a small error in d may lead to a significant difference in the world coordinates [40]. This makes the surface fitting performed in the WCS less accurate than that carried out in the disparity domain. Hence, Mikhailiuk et al. integrated the RANSAC into the road surface fitting to improve the modelling accuracy, where a quadratic surface $d = c_0 + c_1u + c_2v + c_3u^2 + c_4uv + c_5v^2$ is fitted to a set of n random points $[u, v, d]^\top$ in the disparity domain [89] iteratively and the parameters of $\boldsymbol{c}$ are updated when the percentage of the inlier goes higher. Eq. 6.4 can thus be rewritten as follows:

$$\boldsymbol{M}^\top \boldsymbol{M}\boldsymbol{c} = \boldsymbol{M}^\top \boldsymbol{d} \tag{6.12}$$

where

$$\boldsymbol{M} = \begin{bmatrix} 1 & u_1 & v_1 & {u_1}^2 & u_1v_1 & {v_1}^2 \\ 1 & u_2 & v_2 & {u_2}^2 & u_2v_2 & {v_2}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & u_n & v_n & {u_n}^2 & u_nv_n & {v_n}^2 \end{bmatrix} \tag{6.13}$$

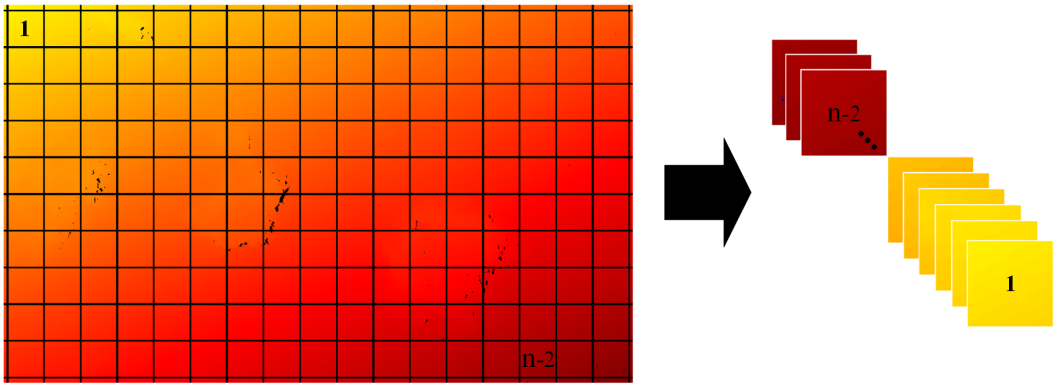


Figure 6.7: Grid on the disparity map.

$$\boldsymbol{d} = [d_1, d_2, \ldots, d_n]^\top \tag{6.14}$$

In this chapter, the author improves on the works presented in [88] and [89] by proposing a robust two-step modelling algorithm where disparities are interpolated into a quadratic surface more accurately. The proposed disparity map modelling algorithm is developed based on the following assumptions:

- the best modelling accuracy corresponds to the highest percentage of inliers.
- the image gradient $\boldsymbol{g^i}$ and the surface gradient $\boldsymbol{g^s}$ of an inlier are almost in the same direction [88].
- the difference between the actual and fitted disparity values of an inlier is small [89].

More details on the modelling procedure are provided in algorithm 12, where $\boldsymbol{c}$ is estimated roughly in the first step and its accuracy is improved iteratively in the second step.

In the first step, the surface fitting is performed in conjunction with the RANSAC. To select a group of points randomly from the disparity map, a grid is first created and the disparity map is divided into a group of square blocks (see Figure 6.7). Then, the percentage $\eta_{\varkappa_d}$ of distress areas in each block is computed. If the value of $\eta_{\varkappa_d}$ exceeds a pre-set tolerance for a given block, the

Algorithm 12: Two-step quadratic surface modelling.

Data: input disparity map ℓ^i, segmentation image $\varkappa$
Result: modelled disparity map ℓ^m

approximate $\boldsymbol{g^i}$ using a gradient filter;
divide ℓ^i into a group of square blocks;
compute $\eta_{\varkappa_d}$ for each block;
if $\eta_{\varkappa_d} > \varepsilon_{\varkappa_d}$ **then**
    ignore the corresponding block when selecting the random points for surface fitting;
end
for *iteration* $\leftarrow$ *1 to N* **do**
    select a random point $\boldsymbol{p}$ from each block;
    fit a quadratic surface to the selected points $\boldsymbol{P}$ and get the parameters of $\boldsymbol{c}$;
    compute $\boldsymbol{g^m}$ for each point in $\boldsymbol{q}$;
    compute θ for each point in $\boldsymbol{q}$;
    compute Δd for each point in $\boldsymbol{q}$;
    determine the number of inliers and outliers: $n_{\mathscr{I}}$ and $n_{\mathscr{O}}$;
    compute the percentage $\eta_{\mathscr{I}}$ of the inliers;
    record $\eta_{\mathscr{I}}$ and the parameters of $\boldsymbol{c}$;
end
select $\boldsymbol{c}$ which corresponds to the highest $\eta_{\mathscr{I}}$;
do
    compute $\boldsymbol{g^m}$ for each point in $\boldsymbol{Q}$;
    compute θ for each point in $\boldsymbol{Q}$;
    compute Δd for each point in $\boldsymbol{Q}$;
    determine $n_{\mathscr{I}}$ and $n_{\mathscr{O}}$;
    remove $\mathscr{O}$ from $\boldsymbol{Q}$;
    fit a quadratic surface to $\boldsymbol{Q}$ and get the updated $\boldsymbol{c}$;
while $n_{\mathscr{O}} \neq \emptyset$;

latter is considered to be unsatisfactory and is therefore omitted from the random sampling process. A random point $\boldsymbol{p} = [u, v, d]^\top$ is then selected from each satisfactory block and a quadratic surface is fitted to these randomly selected points $\boldsymbol{P} = [\boldsymbol{p_1}, \boldsymbol{p_2}, \ldots, \boldsymbol{p_m}]^\top$. The parameters of $\boldsymbol{c}$ can thus be obtained by solving the function in Eq. 6.12. Next, for each point $\boldsymbol{q} = [u, v, d]^\top$ in the disparity map, its surface gradient $\boldsymbol{g^s} = [g^s_u, g^s_v]^\top$ can be estimated from $\boldsymbol{c}$ using Eq. 6.15 and Eq. 6.16 and its image gradient $\boldsymbol{g^i} = [g^i_u, g^i_v]^\top$ can be approximated by

performing a gradient filtering on the disparity map. $\boldsymbol{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_n}]^\top$ is a matrix consisting of all the points in the disparity map. In this chapter, g_u^i and g_v^i are obtained by convolving the disparity map with Sobel horizontal and vertical kernels, respectively.

$$g_u^s = c_1 + 2c_3u + c_4v \tag{6.15}$$

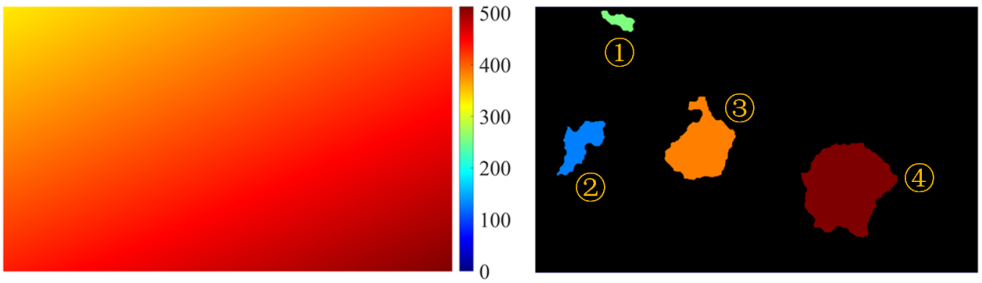
$$g_v^s = c_2 + 2c_4u + c_5v \tag{6.16}$$

The angle θ between the orientations of $\boldsymbol{g^i}$ and $\boldsymbol{g^s}$ can therefore be computed using Eq.6.17. If θ exceeds a pre-set tolerance ε_θ, the corresponding point is classified as an outlier. Furthermore, the difference Δd is computed between the actual and fitted disparity values for the remaining points. If Δd of a point $\boldsymbol{q} = [u, v, d]^\top$ exceeds the threshold ε_d, $\boldsymbol{q}$ is also marked as an outlier.

$$\theta = \arccos\left(\frac{\boldsymbol{g^i} \cdot \boldsymbol{g^s}}{||\boldsymbol{g^i}||_2||\boldsymbol{g^s}||_2}\right) \tag{6.17}$$

The procedures mentioned above iterate N times. In each iteration, the parameters of $\boldsymbol{c}$ and the percentage $\eta_{\mathscr{I}}$ of inliers $\mathscr{I}$ are recorded. After Nth iteration, the author selects the parameters of $\boldsymbol{c}$ which correspond to the highest $\eta_{\mathscr{I}}$ as the coefficients of the quadratic surface.

In the second step, the values of $\boldsymbol{c}$ are updated iteratively. In each iteration, the author determines the inlier set $\mathscr{I}$ and outlier set $\mathscr{O}$ by comparing their θ and Δd. Then, the author removes $\mathscr{O}$ from $\mathscr{Q}$ and fit the quadratic surface to the inliers $\mathscr{I}$. The parameters of $\boldsymbol{c}$ are therefore updated and used in the next



(a) (b)

Figure 6.8: Modelled disparity map and pothole classification. (a) modelled disparity map. (b) pothole classification.

iteration. This process iterates until the number of outliers becomes 0. The modelled disparity map is shown in Figure 6.8a. The evaluation of the proposed two-step disparity map modelling algorithm is discussed in Section 6.2.

6.1.3.2 Disparity Comparison and Post-Processing

By comparing the difference between the actual and modelled disparity maps, a pixel in the pothole areas can be identified if its disparity difference between the two disparity maps is larger than a pre-set threshold δ. Then, the connected components which contain fewer than p pixels are removed because these small objects can severely affect the accuracy of the CCL-based pothole classification. Furthermore, the small holes in the connected components are also filled to ensure the integrity of the potholes. Then, each connected component is labelled as a pothole using the CCL. The classification result is shown in Figure 6.8b, where different colour represents different potholes.

6.1.4 Pothole Tracking

Since the scale of a pothole varies gradually in a sequence of successive frames, the scale adaptive trackers are more desired for pothole tracking. In this chapter, the DSST is utilised to track the detected potholes in successive frames. An

(a) (b)

Figure 6.9: Pothole tracking. (a) tracked pothole in frame 232. (b) tracked pothole in frame 238.

example of the pothole tracking results is illustrated in Figure 6.9, where the size of the tracked target increases as the car approaches the pothole. More details on the DSST is available in [123].

6.2 Experimental Results

In this section, the performance of the proposed pothole detection, classification and tracking system is evaluated both qualitatively and quantitatively. This system is implemented in Matlab 2017b platform on an Intel Core i7-4720HQ CPU using a single thread. The following subsections provide more details on the experimental set-up and performance evaluation.

6.2.1 Experimental Set-Up

In this chapter, a stereo camera from ZED Stereolabs is utilised for data acquisition. More details on the specifications of the ZED stereo camera are available in [108]. An example of the experimental set-up is shown in Figure 6.10.

Using the above experimental set-up, the author created three datasets for pothole detection. Each dataset contains a pair of rectified left and right images, an estimated subpixel disparity map and the corresponding 3-D point cloud date. These datasets are publicly available at http://www.ruirangerfan.com.

Figure 6.10: Experimental set-up.

The following subsections analyse the performance of the proposed system in terms of roll angle estimation, disparity map transformation, disparity map modelling and pothole detection and classification.

6.2.2 Evaluation of Roll Angle Estimation

The author first analyse the computational complexity of the proposed γ estimation algorithm. The computational complexity for estimating γ without using the GSS is $O(\frac{\pi}{\kappa})$, where κ is the step size chosen in the interval $(-\pi/2, \pi/2]$. The GSS reduces the search range exponentially as the interval size becomes only $k^n\pi$ after the nth iteration. This reduces the computational complexity to $O(\log_k \frac{\kappa}{\pi})$. In the experiments, κ is set to $\pi/18000$ (approximately $0.01°$), and the GSS-based γ estimation algorithm only needs to iterate 21 times to get the desirable roll angle with a precision of $\pm 0.01°$.

To evaluate the performance of the proposed γ estimation algorithm in terms of the accuracy, some disparity maps are created according to the values of $\boldsymbol{c}$ which are obtained from the practical experiments. An example of the manually created disparity maps is shown in Figure 6.11a. Then, the disparity map in Figure 6.11a is rotated at different angles γ between $-\pi/3$ and $\pi/3$. In each rotation, a roll angle $\tilde{\gamma}$ is estimated and the absolute error $\Delta\gamma = |\gamma - \tilde{\gamma}|$ between $\tilde{\gamma}$ and its ground truth γ is computed. The relationship between γ and the average of the absolute errors $\Delta\gamma$ is shown in Figure 6.11c, where the maximum absolute error is approximately 8×10^{-5} (around $0.005°$) and the average of the absolute errors is around 1.7×10^{-5}.

Furthermore, the Gaussian white noise $\xi\omega$ is added to the disparity map in Fig. 6.11a for the evaluation of algorithm's robustness to noise, where $\omega \in [-1, 1]$ is a random decimal number satisfying the Gaussian distribution, and ξ is a parameter set to control the intensity of the noise. In the experiments, ξ is increased from 0 to 50 to overestimate the effects caused by noise. The disparity map in Fig. 6.11a with a random noise of 50ω is shown in Figure 6.11b. The average of the absolute errors $\Delta\gamma$ increases by approximately 1.3×10^{-4} with ξ going from 0 to 50. This indicates that the proposed roll angle estimation can ensure high accuracy even when the disparity map is affected by noise.

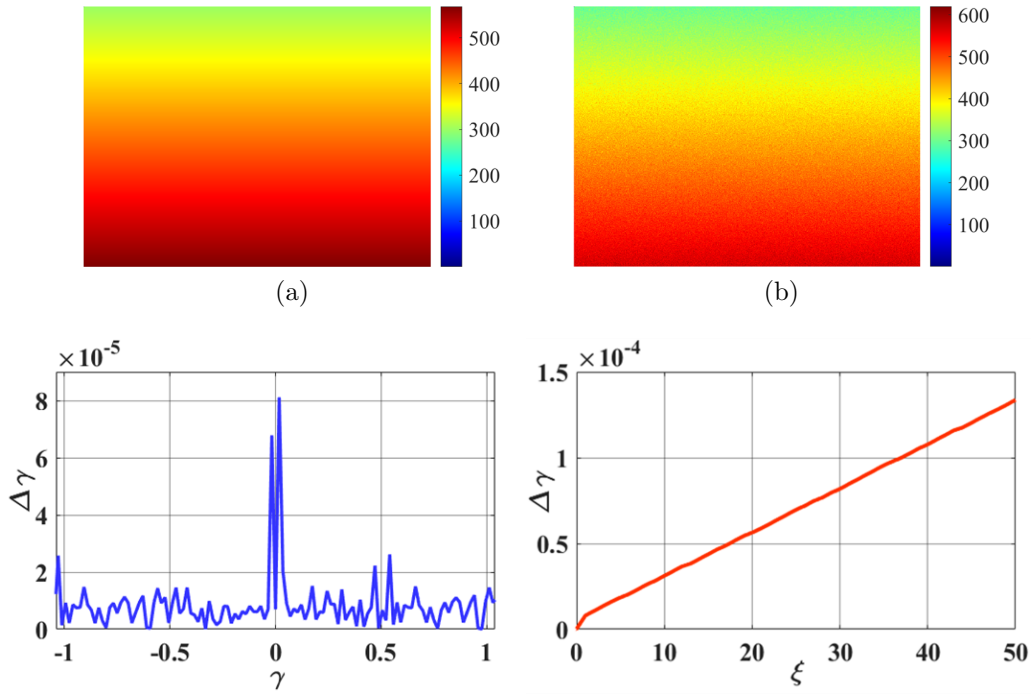


Figure 6.11: Evaluation of roll angle estimation. (a) an example of the manually created disparity maps. (b) the disparity map in (a) with Gaussian white noise ($\xi = 50$). (c) the relationship between different roll angles γ and the average of the absolute errors $\Delta\gamma$. (d) the relationship between different noise intensity control parameter ξ and the average of the absolute errors $\Delta\gamma$.

The author further evaluates algorithm's accuracy using the EISATS synthesised stereo sequences [124, 125], where the roll angle is zero and several vehicles are on the road surface. Some examples of the experimental results are given in Figure 6.12. The average of the absolute errors $\Delta\gamma$ for the EISATS dataset is approximately 0.00647° which is still low and satisfactory.

6.2.3 Evaluation of Disparity Map Transformation

The evaluation of $\boldsymbol{\alpha}$ estimation has been provided in [6], and therefore the author only discusses the performance of disparity map transformation in this subsection. Some examples of the transformed disparity maps are illustrated in Figure 6.12 and Figure 6.15, where the values of δ are set to 3 and 30, respectively. Due to

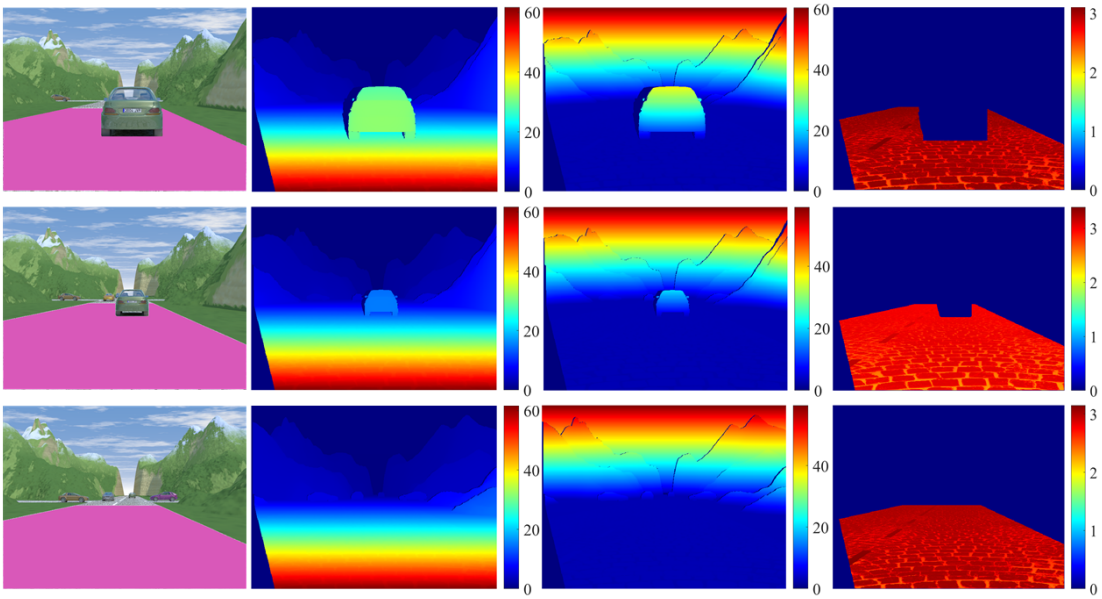


Figure 6.12: Experimental results of EISATS synthesised stereo sequence 1. The first column shows the left images, where areas in magenta are the manually selected road surface. The second column shows the disparity ground truth. The third column shows the transformed disparity maps. The fourth column shows the transformed disparity values of the selected areas.

the fact that the proposed algorithm only aims at transforming the disparities for the road surface, the author selects a region of interest (see the magenta areas in the first column of Figure 6.12) from each image to evaluate the performance of the algorithm for the EISATS dataset.

It can be observed that the disparity values of the road surface areas in the transformed disparity maps become uniform while they differ greatly from those of obstacles and potholes. In Figure 6.12, it can be also observed that the disparity values of vehicles and mountains increase gradually with an increase in v. This occurs because the value of $f(v)$ decreases gradually with a decrease in v and it becomes negative when v becomes smaller than the horizontal coordinate of the vanishing point, as proved in [6].

The non-distress areas can thus be explicitly extracted from the transformed disparity maps by carrying out 2-D Otsu's segmentation algorithm (see the third column of Figure 6.15). The candidates in the distress areas are then omitted from the process of disparity map modelling. The performance improvements

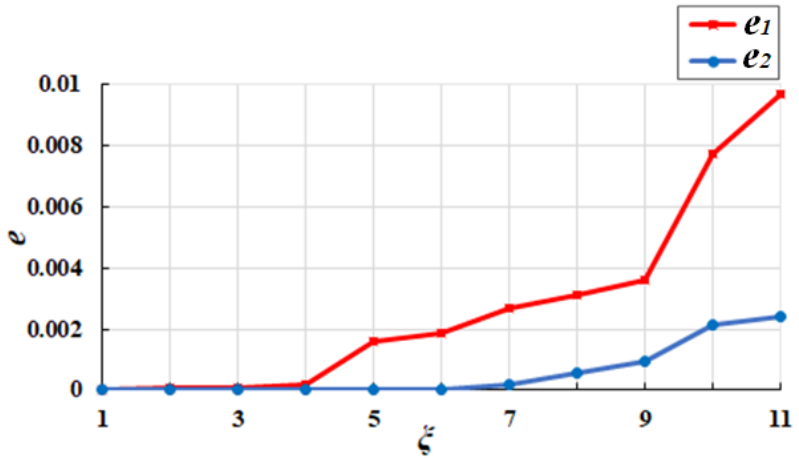


Figure 6.13: Comparison between e_1 and e_2 with respect to different values of ξ.

achieved by ignoring the candidates in distress areas for disparity map modelling will be discussed in the next subsection.

6.2.4 Evaluation of Disparity Map Modelling

In this subsection, the author evaluates the performance of the proposed disparity map modelling algorithm. The disparity maps created in subsection 6.2.2 are utilised as the benchmark data for quantitive analysis. A disparity map without Gaussian white noise is regarded as the ground truth, and the disparity maps with different levels of noise are interpolated into the quadratic surfaces. By comparing the difference between the modelled disparity maps and the ground truth, the average modelling errors e in pixel with respect to different noise intensity can be obtained. A comparison between e_1 and e_2 is shown in Figure 6.13, where e_1 and e_2 represent the average disparity modelling errors after the first and second steps, respectively. From Figure 6.13, it can be seen that the accuracy of the modelled disparity maps using either the first step or both steps becomes lower with an increase in ξ. Also, by iteratively updating the inliers and the parameters of $\boldsymbol{c}$, the accuracy of the modelled disparity map is improved using two steps.

Then, the author compares the precision of the quadratic surface fitting between several state-of-the-art road surface 3-D modelling algorithms and the two-step disparity map modelling algorithm proposed in subsection 6.1.3.1. Since the

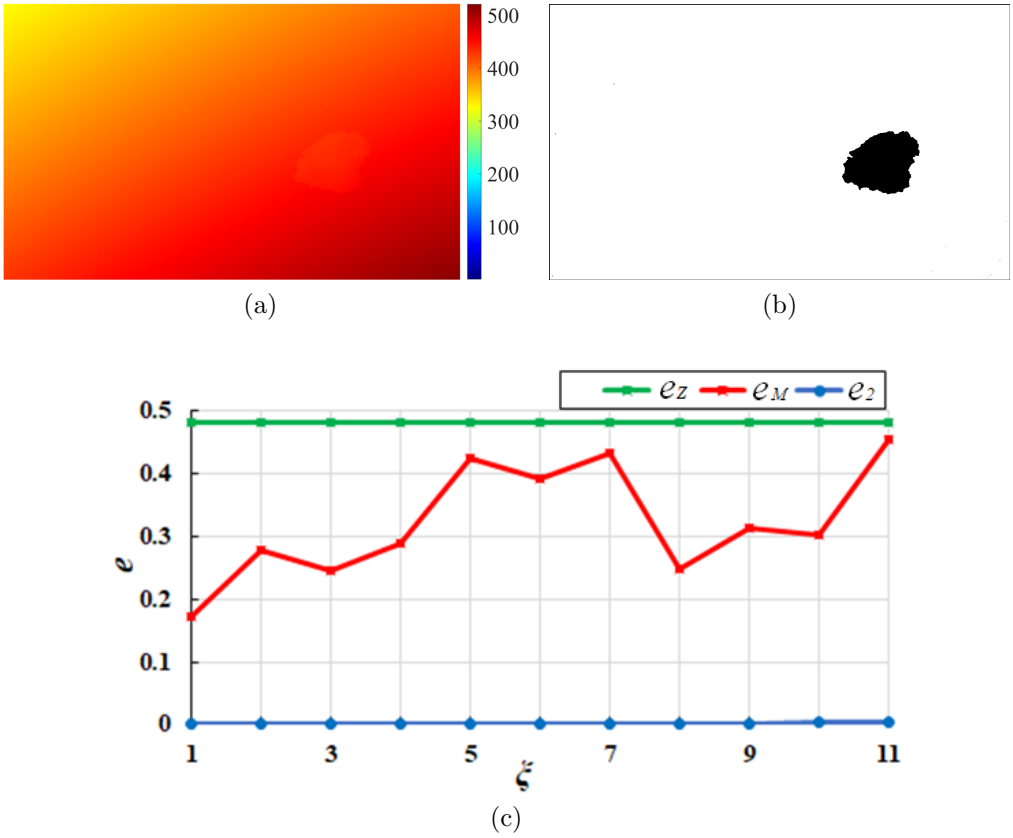


Figure 6.14: Evaluation of the proposed two-step disparity map modelling algorithm. (a) disparity map with a simulated pothole. (b) the corresponding non-distress area extraction result of (a). (c) the comparison among e_Z, e_M and e_2 with respect to different levels of Gaussian white noise.

optimisation solution in algorithm [6] is not always valid, the proposed two-step disparity map modelling algorithm is only compared with the algorithms described in [87] and [89]. According to the disparity difference of the potholes between the actual and modelled disparity maps, the author first created several disparity maps with simulated potholes as the benchmark data. An example of these disparity map is shown in Figure 6.14a. Next, the author defines two notations e_Z and e_M as the average modelling errors obtained when using the algorithms [87] and [89], respectively. By transforming the disparity maps using the algorithm proposed in subsection 6.1.1.2, the distress areas become more distinguishable and they can be explicitly extracted from the transformed disparity map using Otsu's segmentation algorithm, as shown in Figure 6.14b. However,

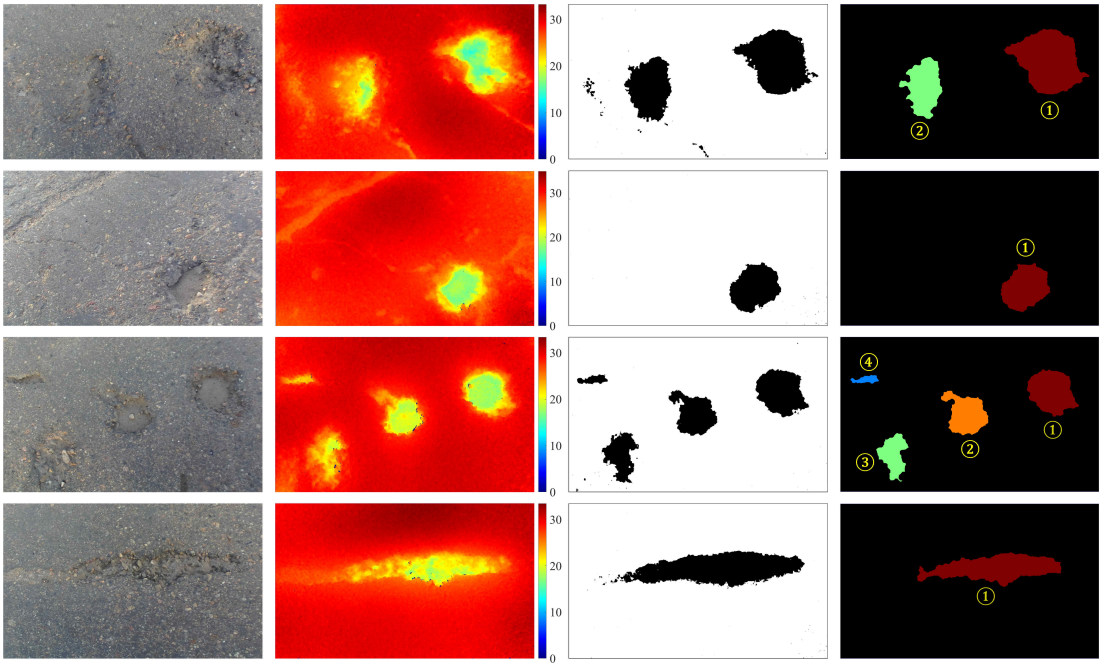
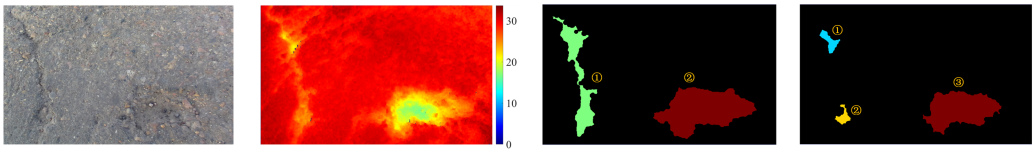


Figure 6.15: Experimental results of pothole detection and classification. The first column shows the left images. The second column shows the transformed disparity maps. The third column shows the extracted non-distress areas. The fourth column shows the classification of the detected potholes.

the algorithm in [87] considers every pixel for surface fitting and therefore the average modelling error e_Z is significantly high. In [89], the authors use the RANSAC to reduce the effects caused by the outliers. Although the modelling accuracy has been significantly improved, the pixels which are either in the distress areas or have very different gradients from the expected ones still severely affect the accuracy of the disparity map modelling. The comparison among e_Z, e_M and e_2 with respect to different levels of Gaussian white noise is illustrated in Figure 6.14c. Compared with the algorithm proposed in [89], the modelling accuracy achieved using the proposed two-step modelling algorithm is increased by around 189 times when ξ is set to 11.



(a) (b) (c) (d)

Figure 6.16: An example of pothole detection and classification results using different parameters. (a) the left images. (b) the transformed disparity map of (a). (c) the pothole classification result when $\delta = 2.8$ and $p = 100$. (d) the pothole classification result when $\delta = 5$ and $p = 100$.

6.2.5 Evaluation of Pothole Detection and Classification

Since the disparity map can be represented by a quadratic surface, the problem of detecting potholes can thus be solved by comparing the difference between the actual and modelled disparity maps and identifying the pixel whose disparity difference exceeds the pre-set threshold δ. The subsection 6.2.4 shows that the proposed algorithm in this chapter can accurately model the disparity map as a quadratic surface. Therefore, the pothole detection and classification results entirely depend on the two parameters δ and p which are set to extract the potential pothole areas and remove the small connected components, respectively. Due to the fact that the actual environment is usually complicated, it is difficult to choose an optimum parameter which can be applied to every scenario. An example of pothole detection and classification results with different parameters is shown in Figure 6.16, where different values of δ may lead to incorrect detections. To achieve the best detection accuracy, the values of δ and p are set to 4.5 and 100, respectively. The detection results of the created datasets are shown in table 6.1, where the total successful detection rate is approximately 99%. Furthermore, The execution of the proposed pothole detection and classification algorithm in

Table 6.1: Detection results of the proposed algorithm.

Dataset	Total potholes	Correct detection	Incorrect detection	Misdetection
Dataset 1	22	22	0	0
Dataset 2	52	51	1	0
Dataset 3	5	5	0	0

Matlab 2017b takes around 2.45s. Although the proposed algorithm does not run in real-time, the author believes that its speed can be increased in the future by exploiting the parallel computing architecture.

6.3 Conclusion

The main novelties of this chapter include a disparity map transformation algorithm and a two-step disparity map modelling algorithm. The distress areas become more distinguishable in the transformed disparity map and can thus be extracted explicitly using 2-D Otsu's thresholding algorithm. This greatly improves the accuracy of the disparity map modelling. To achieve a better processing efficiency, the GSS and DP are utilised to estimate the transformation parameters. Then, the disparity map is interpolated into a quadratic surface using the proposed two-step disparity map modelling algorithm. By ignoring the points in distress areas and integrating the gradient information into the surface fitting, the accuracy of the modelled disparity map is significantly improved. The potholes can then be detected by comparing the difference between the actual and modelled disparity maps and labelling the connected components whose disparity difference exceeds a pre-set threshold. The detected potholes are also tracked using the DSST. The experimental results show that the proposed algorithm can estimate the roll angle and model the quadratic surface accurately. The overall successful detection rate is around 99% .

Chapter 7

Conclusions

7.1 Thesis Summary

In this thesis, the author presented the real-time implementation of an efficient disparity estimation algorithm for three stereo vision-based automotive applications, i.e., lane detection, road surface 3-D reconstruction and pothole detection.

In Chapter 2, the author first provided some basic but important concepts of computer stereo vision and multiple view geometry. Then, the literature reviews of lane detection, road surface 3-D reconstruction and pothole detection were covered, respectively. Finally, the author provided a general description of heterogeneous systems.

In Chapter 3, the implementation of a disparity estimation algorithm was detailed, where the stereo matching was optimised by factorising the NCC equation into five independent parts and their computations are accelerated using four integral images. The implementation exploits the parallel computing architecture and a real-time performance has been achieved on both a multi-threading CPU and a GPU. Compared with other subsystems, e.g., lane detection and pothole detection, in the ADAS, stereo vision usually accounts for a big chunk of the whole processing time. Therefore, the real-time stereo vision implementation proposed in this chapter allows for more room in terms of processing time for the other subsystems in the ADAS. The main contributions of this chapter are published in [11].

The estimated disparity maps in Chapter 4 were then utilised to improve the

robustness of a multiple lane detection system, where the lanes were modelled using dense vanishing points. The latter was estimated using the information of both disparity and gradient. To further improve the process of dense vanishing point estimation, the RANSAC was utilised to update the parameters of the road model iteratively until the percentage of the inliers exceeded a pre-set threshold. Furthermore, the author proposed a novel lane position validation method which computes the energy of each possible solution and selects all satisfying lane positions for visualisation. The proposed lane detection algorithm was implemented on a heterogeneous system which consists of an Intel Core i7-4720HQ CPU and an NVIDIA GTX 970M GPU and a processing speed of 143 fps has been achieved. The proposed lane detection system is capable of detecting multiple lane markings in real time. This can enhance the driving safety and reduce the number of fatalities on the road. More details on the proposed lane detection algorithm are provided in [6].

As for the road surface 3-D reconstruction algorithm described in Chapter 5, the main novelties include a perspective transformation algorithm, a correlation maxima verification approach and a disparity map global refinement strategy. The perspective transformation not only enhances the similarity of a ground plane between two images but also reduces the search range for stereo matching. This makes the SRP stereo perform more accurately and efficiently. The correlation maxima verification further offsets the insufficient propagation in the SRP stereo and guarantees the feasibility of parabola interpolation in the subpixel enhancement phase. By iteratively minimising the energy with respect to the interpolated parabolas, the subpixel disparity map was optimised. The disparities in a continuous area become smoother, but they are preserved when discontinuities occur. The maximal absolute error of the 3-D reconstruction is around 3 mm, which satisfies the requirement of millimetre accuracy for on-road damage detection. Also, the author created three datasets to contribute to the stereo vision-based road surface 3-D reconstruction and made them publicly available at https://github.com/ruirangerfan/road_surface_3d_reconstruction_datasets. The proposed algorithm in this chapter provides an alternative to laser scanners for 3-D road surface reconstruction. The stereo cameras are more cost-effective and easier for maintenance than the laser scanning equipments. This also provides

an easier way to assess the road condition in the long term by mapping the 3-D road surface across the city and building a database.

Finally, the author presented a complete system for pothole detection, classification and tracking in Chapter 6. The main contributions include a novel disparity map transformation algorithm and a two-step disparity map modelling algorithm. The input dense disparity maps were first transformed to better distinguish the distress areas. The disparities in the non-distress areas were then interpolated into a quadratic surface for pothole detection, where the gradient information was integrated into the surface fitting and the parameters of the quadratic surface were updated iteratively until the percentage of the inliers exceeded a pre-set threshold. Then, the detected potholes were tracked using the DSST. Similar to the road surface 3-D reconstruction system presented in Chapter 5, the author created three synthetic pothole datasets using a ZED stereo camera, which are also publicly available. The proposed system in this chapter can help the certified inspectors and structural engineers to identify the potholes more accurately and efficiently. Furthermore, it not only ensures the safety of the personnel but also reduces the time for detecting the potholes all around the city. Moreover, the detection results become less subjective because they no longer entirely depend on the experience of the personnel [81].

7.2 Future Work

In Chapter 3 and Chapter 5, the proposed algorithms are not able to fully exploit the parallel computing architecture of the graphics cards to estimate disparity maps. This is because the propagation strategy used is not efficient enough and can not be adapted for different platforms. One way to improve on that is to coarsely estimate the disparity map in the first iteration which is then optimised iteratively until a finer estimate is obtained. Also, since the feature extraction algorithms, such as SIFT [126], SURF [127] and BRISK [104], can extract keypoints from images and match the correspondence pairs between two different frames, these can be utilised to provide some confidential seeds in the initial stage of disparity map estimation. Furthermore, errors in stereo calibration always affect the precision of the stereo matching significantly. In this regard,

a self-calibration algorithm can be designed to enhance the robustness of the proposed stereo vision system, where the essential matrix is estimated from a set of correspondence pairs and is then optimised iteratively to provide a better accuracy using the bundle adjustment.

As discussed in Chapter 4, some actual road conditions may result in failed detections. To tackle this issue, a deep neural network can be trained for dense vanishing point estimation and lane position validation. Furthermore, in order to yield better efficiency, accuracy and robustness, the proposed lane detection algorithm in Chapter 4 can be implemented using some state-of-the-art embedded systems, such as Jetson TK2 from NVIDIA.

Additionally, since the proposed road surface 3-D reconstruction algorithm in Chapter 5 has achieved some highly precise point clouds, it can be applied to road surface SLAM (Simultaneous Localisation and Mapping) for smart city applications.

In Chapter 6, the parameters set to detect the pothole areas cannot be applied to all cases. Hence, a deep neural network can be trained to detect potholes directly from the transformed disparity maps. Moreover, as discussed in Chapter 6, the disparity map transformation algorithm which is developed based on golden section search and dynamic programming still needs to iterate many times to get an accurate transformation parameter. This can be improved by estimating the transformation parameter using a two-step method, where the resolution of the input disparity map is reduced and a coarse transformation parameter is first estimated from the low-resolution disparity map. Then, the transformation parameter is refined iteratively by performing the least squares fitting on the original disparity map.

Bibliography

[1] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5695–5703.

[2] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3354–3361.

[3] *SABRE-SCAN*, SABRE, http://www.sabresurvey.com/sabre-scan.html, accessed: 2018-03-29. [Online]. Available: http://www.sabresurvey.com/sabre-scan.html

[4] B. Russell, *Microsoft's Kinect is officially dead*, technobuffalo, accessed: 2017-10-25. [Online]. Available: https://www.technobuffalo.com/2017/10/25/microsofts-kinect-is-officially-dead/

[5] A. Andreas, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[6] U. Ozgunalp, R. Fan, X. Ai, and N. Dahnoun, "Multiple lane detection algorithm based on novel dense vanishing point estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 3, pp. 621–632, 2017.

[7] N. Qian, "Binocular disparity and the perception of depth," *Neuron*, vol. 18, no. 3, pp. 359–368, 1997.

[8] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[9] F. Sadjadi and E. Ribnick, "Passive 3d sensing, and reconstruction using multi-view imaging," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE, 2010, pp. 68–74.

[10] T. Emanuele and V. Alessandro, "Introductory techniques for 3-d computer vision," 1998.

[11] R. Fan and N. Dahnoun, "Real-time implementation of stereo vision based on optimised normalised cross-correlation and propagated search range on a gpu," in *Imaging Systems and Techniques (IST), 2017 IEEE International Conference on*. IEEE, 2017, pp. 241–246.

[12] B. J. Tippetts, *Real-Time Stereo Vision for Resource Limited Systems*. Brigham Young University, 2012.

[13] B. Tippetts, D. J. Lee, K. Lillywhite, and J. Archibald, "Review of stereo vision algorithms and their suitability for resource-limited systems," *Journal of Real-Time Image Processing*, vol. 11, no. 1, pp. 5–25, 2016.

[14] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[15] A. T. Ihler, W. F. John III, and A. S. Willsky, "Loopy belief propagation: Convergence and effects of message errors," *Journal of Machine Learning Research*, vol. 6, no. May, pp. 905–936, 2005.

[16] M. F. Tappen and W. T. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters," in *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE, 2003, p. 900.

[17] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[18] M. G. Mozerov and J. van de Weijer, "Accurate stereo matching by two-step energy minimization," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 1153–1163, 2015.

[19] S. N. Sinha, D. Scharstein, and R. Szeliski, "Efficient high-resolution stereo matching using local plane sweeps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1582–1589.

[20] M. Bleyer, C. Rhemann, and C. Rother, "Extracting 3d scene-consistent object proposals and depth from stereo images," *Computer Vision–ECCV 2012*, pp. 467–481, 2012.

[21] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *European Conference on Computer Vision*. Springer, 2014, pp. 756–771.

[22] R. Sara, "Finding the largest unambiguous component of stereo matching," *Computer Vision-ECCV 2002*, pp. 900–914, 2002.

[23] R. Sara, R., "Robust correspondence recognition for computer vision," in *Compstat 2006-Proceedings in Computational Statistics*. Springer, 2006, pp. 119–131.

[24] J. Cech and R. Sara, "Efficient sampling of disparity space for fast and accurate matching," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[25] R. Spangenberg, T. Langner, and R. Rojas, "Weighted semi-global matching and center-symmetric census transform for robust driver assistance," in *International Conference on Computer Analysis of Images and Patterns*. Springer, 2013, pp. 34–41.

[26] O. Miksik, Y. Amar, V. Vineet, P. Pérez, and P. H. Torr, "Incremental dense multi-modal 3d scene reconstruction," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on.* IEEE, 2015, pp. 908–915.

[27] S. Pillai, S. Ramalingam, and J. J. Leonard, "High-performance and tunable stereo reconstruction," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on.* IEEE, 2016, pp. 3188–3195.

[28] Z. Zhang, X. Ai, and N. Dahnoun, "Efficient disparity calculation based on stereo vision with ground obstacle assumption," in *21st European Signal Processing Conference (EUSIPCO 2013).* IEEE, 2013, pp. 1–5.

[29] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on.* IEEE, 2015, pp. 4353–4361.

[30] J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1592–1599.

[31] ——, "Stereo matching by training a convolutional neural network to compare image patches," *Journal of Machine Learning Research*, vol. 17, no. 1-32, p. 2, 2016.

[32] D. Scharstein and R. Szeliski, "Middlebury stereo vision page," 2002.

[33] ——, "Middlebury stereo vision and evaluation page," 2005.

[34] ——, "Middlebury stereo evaluation-version 2," *The Middlebury Computer Vision Pages (online), available from (accessed 2015-03-02)*, 2011.

[35] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[36] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3061–3070.

[37] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "The kitti vision benchmark suite," 2015.

[38] M. Menze, C. Heipke, and A. Geiger, "Joint 3d estimation of vehicles and scene flow," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 2, p. 427, 2015.

[39] Z. Zhang, "Advanced stereo vision disparity calculation and obstacle analysis for intelligent vehicles," Ph.D. dissertation, University of Bristol, 2013.

[40] R. Fan, X. Ai, and N. Dahnoun, "Road surface 3D reconstruction based on dense subpixel disparity map estimation," *IEEE Transactions on Image Processing*, vol. PP, no. 99, p. 1, 2018.

[41] J. A. Brink, R. L. Arenson, T. M. Grist, J. S. Lewin, and D. Enzmann, "Bits and bytes: the future of radiology lies in informatics and information technology," *European Radiology*, pp. 1–5, 2017.

[42] R. Fan, V. Prokhorov, and N. Dahnoun, "Faster-than-real-time linear lane detection implementation using soc dsp tms320c6678," in *Imaging Systems and Techniques (IST), 2016 IEEE International Conference on.* IEEE, 2016, pp. 306–311.

[43] K. Joreskog and R. Reymont, "Applied factor analysis in the natural sciences," 1993.

[44] J. Weng, P. Cohen, M. Herniou *et al.*, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 10, pp. 965–980, 1992.

[45] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[46] D. Brown, "Lens distortion for close-range photogrammetry," *Photometric Engineering*, vol. 37, no. 8, pp. 855–866, 1971.

[47] D. C. Brown, "Decentering distortion of lenses," *Photogrammetric Engineering and Remote Sensing*, 1966.

[48] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, pp. 133–135, 1981.

[49] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 6, pp. 580–593, 1997.

[50] Z. Hu, F. Lamosa, and K. Uchimura, "A complete uv-disparity study for stereovision based 3d driving environment analysis," in *Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05)*. IEEE, 2005, pp. 204–211.

[51] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 9, pp. 1582–1599, 2009.

[52] A. Hosni, M. Bleyer, and M. Gelautz, "Secrets of adaptive support weight techniques for local stereo matching," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 620–632, 2013.

[53] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on*. IEEE, 1998, pp. 839–846.

[54] Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 492–504, 2009.

[55] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," *IEEE Transactions*

on Pattern Analysis and Machine Intelligence, vol. 35, no. 2, pp. 504–511, 2013.

[56] Z. Zhang, X. Ai, N. Canagarajah, and N. Dahnoun, "Local stereo disparity estimation with novel cost aggregation for sub-pixel accuracy improvement in automotive applications," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*. IEEE, 2012, pp. 99–104.

[57] A. Blake, P. Kohli, and C. Rother, *Markov random fields for vision and image processing*. Mit Press, 2011.

[58] S. Z. Li, *Markov random field modeling in computer vision*. Springer Science & Business Media, 2012.

[59] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *International journal of computer vision*, vol. 12, no. 1, pp. 43–77, 1994.

[60] S. P. Narote, P. N. Bhujbal, A. S. Narote, and D. M. Dhane, "A review of recent advances in lane detection and departure warning system," *Pattern Recognition*, vol. 73, pp. 216–234, 2018.

[61] M. Bertozzi and A. Broggi, "Gold: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE transactions on image processing*, vol. 7, no. 1, pp. 62–81, 1998.

[62] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using b-snake," *Image and Vision computing*, vol. 22, no. 4, pp. 269–280, 2004.

[63] U. Ozgunalp and N. Dahnoun, "Robust lane detection & tracking based on novel feature extraction and lane categorization," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 8129–8133.

[64] K. Kluge and S. Lakshmanan, "A deformable-template approach to lane detection," in *Intelligent Vehicles' 95 Symposium., Proceedings of the*. IEEE, 1995, pp. 54–59.

[65] Y. Wang, L. Bai, and M. Fairhurst, "Robust road modeling and tracking using condensation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 4, pp. 570–579, 2008.

[66] Y. Zhou, R. Xu, X. Hu, and Q. Ye, "A robust lane detection and tracking method based on computer vision," *Measurement science and technology*, vol. 17, no. 4, p. 736, 2006.

[67] C. Kreucher and S. Lakshmanan, "Lana: a lane extraction algorithm that uses frequency domain features," *IEEE Transactions on Robotics and automation*, vol. 15, no. 2, pp. 343–350, 1999.

[68] C. R. Jung and C. R. Kelber, "An improved linear-parabolic model for lane following and curve detection," in *Computer Graphics and Image Processing, 2005. SIBGRAPI 2005. 18th Brazilian Symposium on.* IEEE, 2005, pp. 131–138.

[69] Y. Wang, D. Shen, and E. K. Teoh, "Lane detection using spline model," *Pattern Recognition Letters*, vol. 21, no. 8, pp. 677–689, 2000.

[70] M. Nieto, L. Salgado, F. Jaureguizar, and J. Cabrera, "Stabilization of inverse perspective mapping images based on robust vanishing point estimation," in *Intelligent Vehicles Symposium, 2007 IEEE.* IEEE, 2007, pp. 315–320.

[71] D. Schreiber, B. Alefs, and M. Clabian, "Single camera lane detection and tracking," in *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE.* IEEE, 2005, pp. 302–307.

[72] D. Hanwell and M. Mirmehdi, "Detection of lane departure on high-speed roads." in *ICPRAM (2)*, 2012, pp. 529–536.

[73] B. Fardi and G. Wanielik, "Hough transformation based approach for road border detection in infrared images," in *Intelligent Vehicles Symposium, 2004 IEEE.* IEEE, 2004, pp. 549–554.

[74] Y. Wang, N. Dahnoun, and A. Achim, "A novel system for robust lane detection and tracking," *Signal Processing*, vol. 92, no. 2, pp. 319–334, 2012.

[75] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation," in *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 2. IEEE, 2002, pp. 646–651.

[76] E. Schnebele, B. Tanyu, G. Cervone, and N. Waters, "Review of remote sensing methodologies for pavement management and assessment," *European Transport Research Review*, vol. 7, no. 2, pp. 1–19, 2015.

[77] T. Kim and S.-K. Ryu, "Review and analysis of pothole detection methods," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 5, no. 8, pp. 603–608, 2014.

[78] L. Cruz, L. Djalma, and V. Luiz, "Kinect and rgbd images: Challenges and applications graphics," in *2012 25th SIBGRAPI Conference on Patterns and Images Tutorials (SIBGRAPI-T)*, 2012.

[79] C. Koch and I. Brilakis, "Pothole detection in asphalt pavement images," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 507–515, 2011.

[80] E. Buza, S. Omanovic, and A. Huseinovic, "Pothole detection with image processing and spectral clustering," in *Proceedings of the 2nd International Conference on Information Technology and Computer Networks*, 2013, pp. 48–53.

[81] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[82] S. Li, C. Yuan, D. Liu, and H. Cai, "Integrated processing of image and gpr data for automated pothole detection," *Journal of computing in civil engineering*, vol. 30, no. 6, p. 04016015, 2016.

[83] Y.-C. Tsai and A. Chatterjee, "Pothole detection and classification using 3d technology and watershed method," *Journal of Computing in Civil Engineering*, vol. 32, no. 2, p. 04017078, 2017.

[84] M. R. Jahanshahi, F. Jazizadeh, S. F. Masri, and B. Becerik-Gerber, "Unsupervised approach for autonomous pavement-defect detection and quantification using an inexpensive depth sensor," *Journal of Computing in Civil Engineering*, vol. 27, no. 6, pp. 743–754, 2012.

[85] Z. Zhang, Y. Wang, J. Brand, and N. Dahnoun, "Real-time obstacle detection based on stereo vision for automotive applications," in *Education and Research Conference (EDERC), 2012 5th European DSP*. IEEE, 2012, pp. 281–285.

[86] A. Barsi, I. Fi, T. Lovas, G. Melykuti, B. Takacs, C. Toth, and Z. Toth, "Mobile pavement measurement system: A concept study," in *Proc. ASPRS Annual Conference, Baltimore*, 2005, p. 8.

[87] Z. Zhang, X. Ai, C. Chan, and N. Dahnoun, "An efficient algorithm for pothole detection using stereo vision," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 564–568.

[88] U. Ozgunalp, X. Ai, and N. Dahnoun, "Stereo vision-based road estimation assisted by efficient planar patch calculation," *Signal, Image and Video Processing*, vol. 10, no. 6, pp. 1127–1134, 2016.

[89] A. Mikhailiuk and N. Dahnoun, "Real-time pothole detection on tms320c6678 dsp," in *Imaging Systems and Techniques (IST), 2016 IEEE International Conference on*. IEEE, 2016, pp. 123–128.

[90] S. M. Abbas and A. Muhammad, "Outdoor rgb-d slam performance in slow mine detection," in *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*. VDE, 2012, pp. 1–6.

[91] R. Chandra, *Parallel programming in OpenMP*. Morgan kaufmann, 2001.

[92] NVIDIA, "Cuda c programming guide," September 2017. [Online]. Available: http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

[93] A. Munshi, B. Gaster, T. G. Mattson, and D. Ginsburg, *OpenCL programming guide*. Pearson Education, 2011.

[94] R. Fan and N. Dahnoun, "Real-time stereo vision-based lane detection system," *Measurement Science and Technology*, 2018.

[95] C. Lin, Y. Li, G. Xu, and Y. Cao, "Optimizing zncc calculation in binocular stereo matching," *Signal Processing: Image Communication*, vol. 52, pp. 64–73, 2017.

[96] G. Facciolo, N. Limare, and E. Meinhardt-Llopis, "Integral images for block matching," *Image Processing On Line*, vol. 4, pp. 344–369, 2014.

[97] J. P. Lewis, "Fast template matching," in *Vision interface*, vol. 95, no. 120123, 1995, pp. 15–19.

[98] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[99] C. Rafael Gonzalez and R. Woods, "Digital image processing," *Pearson Education*, 2002.

[100] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[101] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine vision and applications*, vol. 25, no. 3, pp. 727–745, 2014.

[102] H. Hattori and A. Maki, "Stereo without depth search and metric calibration," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 1. IEEE, 2000, pp. 177–184.

[103] H. Nakai, N. Takeda, H. Hattori, Y. Okamoto, and K. Onoguchi, "A practical stereo scheme for obstacle detection in automotive use," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3. IEEE, 2004, pp. 346–350.

[104] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*. IEEE, 2011, pp. 2548–2555.

[105] I. Haller and S. Nedevschi, "Design of interpolation functions for subpixel-accuracy stereo-vision systems," *IEEE Transactions on image processing*, vol. 21, no. 2, pp. 889–898, 2012.

[106] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for markov random fields with smoothness-based priors," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 6, pp. 1068–1080, 2008.

[107] G. G. Slabaugh, "Computing euler angles from a rotation matrix," *Retrieved on August*, vol. 6, no. 2000, pp. 39–63, 1999.

[108] *Stereolabs Products*, STEREOLABS, accessed: May 29, 2017. [Online]. Available: https://www.stereolabs.com/zed/specs/. [Online]. Available: https://player.fm/series/tech-tent-business-and-technology/the-race-for-driverless-rides

[109] D. F. Llorca, M. A. Sotelo, I. Parra, M. Ocaña, and L. M. Bergasa, "Error analysis in a stereo vision-based pedestrian detection sensor for collision avoidance applications," *Sensors*, vol. 10, no. 4, pp. 3741–3758, 2010.

[110] S. Mathavan, K. Kamal, and M. Rahman, "A review of three-dimensional imaging technologies for pavement distress detection and measurements," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2353–2362, 2015.

[111] X. Ai, Y. Gao, J. G. Rarity, and N. Dahnoun, "Obstacle detection using u-disparity on quadratic road surfaces," in *Proc. 16th Int. IEEE Conf. Intelligent Transportation Systems (ITSC 2013)*, Oct. 2013, pp. 1352–1357.

[112] J. Ryu, E. J. Rossetter, and J. C. Gerdes, "Vehicle sideslip and roll parameter estimation using gps," in *Proceedings of the AVEC International Symposium on Advanced Vehicle Control*, 2002, pp. 373–380.

[113] J. Ryu and J. C. Gerdes, "Estimation of vehicle roll and road bank angle," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 3. IEEE, 2004, pp. 2110–2115.

[114] H. Eric Tseng, L. Xu, and D. Hrovat, "Estimation of land vehicle roll and pitch angles," *Vehicle System Dynamics*, vol. 45, no. 5, pp. 433–443, 2007.

[115] J. Oh and S. B. Choi, "Vehicle roll and pitch angle estimation using a cost-effective six-dimensional inertial measurement unit," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 227, no. 4, pp. 577–590, 2013.

[116] M. Schlipsing, J. Schepanek, and J. Salmen, "Video-based roll angle estimation for two-wheeled vehicles," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 876–881.

[117] M. Schlipsing, J. Salmen, B. Lattke, K. G. Schröter, and H. Winner, "Roll angle estimation for motorcycles: Comparing video and inertial sensor approaches," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*. IEEE, 2012, pp. 500–505.

[118] R. Labayrade and D. Aubert, "A single framework for vehicle roll, pitch, yaw estimation and obstacles detection by stereovision," in *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*. IEEE, 2003, pp. 31–36.

[119] P. Skulimowski, M. Owczarek, and P. Strumiłło, "Ground plane detection in 3d scenes for an arbitrary camera roll rotation through v-disparity representation."

[120] P. Pedregal, *Introduction to optimization.* Springer Science & Business Media, 2006, vol. 46.

[121] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[122] L. Jianzhuang, L. Wenqing, and T. Yupeng, "Automatic thresholding of gray-level pictures using two-dimension otsu method," in *Circuits and Systems, 1991. Conference Proceedings, China., 1991 International Conference on.* IEEE, 1991, pp. 325–327.

[123] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 8, pp. 1561–1575, 2017.

[124] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn, "Differences between stereo and motion behaviour on synthetic and real-world stereo sequences," in *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference.* IEEE, 2008, pp. 1–6.

[125] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers, "Efficient dense scene flow from sparse or dense stereo data," in *European conference on computer vision.* Springer, 2008, pp. 739–751.

[126] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[127] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision.* Springer, 2006, pp. 404–417.